Distributed Control and Management of Context-Based Wireless Mesh Networks

Ricardo Matos and Susana Sargento

Instituto de Telecomunicações, Universidade de Aveiro, Aveiro, Portugal
{ricardo.matos,susana}@ua.pt

Abstract. Using the flexibility of Wireless Mesh Networks (WMNs), we provide personalized access for highly dynamic mesh clients by splitting a WMN into several logical networks, each one configured to meet a set of specific levels of users' context demands (context can span from security, mobility, cost, services' requirements). In such approach, users can be grouped according to similarity of their context, and can be associated to the logical networks matching their context, built through virtualization (Virtual Networks - VNs). To break the traditional centralized architectures for the control of nodes and networks, this paper defines a novel context-aware distributed control framework to allow users' associations to fitting VNs, and to create, extend, or remove VNs on-demand to be adapted to the dynamics of WMN environments and mesh clients. Moreover, WMN nodes are endowed with autonomous capabilities that allow them to co-operatively control VN topologies based on indicators of resource availability and users' perceived Quality-of-Experience (QoE).

Keywords: Wireless Mesh Networks, Context-Awareness, Network Virtualization, Personalized Connectivity, Distributed Control.

1 Introduction and Related Work

The flexibility and self-properties of Wireless Mesh Networks (WMNs) [2] can be exploited to provide support for new communication and architectural paradigms. One of these paradigms can be the personalization of users' connectivity through a proper selection and configuration of WMN connections, nodes, and transport paths according to users' context [7] (e.g., mobility patterns, security and cost preferences, services' Quality-of-Service (QoS) requirements).

With the context-based WMNs in mind, we proposed in [5] an architecture to split a WMN infrastructure in a set of logical networks, each one configured (in terms of running protocols, assigned resources, and topology) to meet a set of context levels of users. Users are then associated to fitting logical networks, increasing their perceived Quality-of-Experience (QoE). Several literature approaches (e.g., [9][10]) aim to increase the WMN performance by means of context-aware mechanisms; however, they do not consider the concept of splitting the WMN in a set of context-based logical networks.

When building logical networks, network virtualization [6] is considered as a powerful tool to allow a flexible and programmable utilization of these logical

K. Pentikousis et al. (Eds.): MONAMI 2011, LNICST 97, pp. 158–173, 2012.

networks (or Virtual Networks - VNs). There are several solutions (e.g.,[1][8]) used to build multiple VNs over a physical WMN [11][3], but they do not provide mechanisms to adapt VNs to the dynamics of WMN environments.

In [4], we evaluated, through analytical and simulation tools, the impact of applying network virtualization in our approach, and a basic control mechanism to associate users to exactly fitting VNs and to configure VN topologies. In this paper, by distributing the control knowledge and intelligence among the different WMN nodes and their supported VN nodes, we propose a structured-based distributed framework based on the co-operation of these elements to: *(i)* perform context-driven discovery of fitting VNs for users; *(ii)* create, update, extend, and remove VNs on-demand to serve the multitude of users based on context-, topology-, resource-, and QoE-aware metrics. We also detail the control processes to setup, maintain, and update the proposed distributed framework to be adapted to context changing and mobility of mesh clients.

In this paper, Section 2 presents the context-aware WMN architecture, its raised challenges, and a model definition. Then, Section 3 presents the mapping of user's demands in VNs' features, and the metric to select best fitting VNs for users. Section 4 proposes a distributed framework to control the context-aware VNs, which is used in section 5 to associate users to fitting VNs and to manage VN topologies on-demand. Finally, Section 6 concludes the paper and proposes several guidelines for future work.

2 Context-Based Wireless Mesh Networks

This section presents and formally defines the proposed architecture to build context-based WMNs, summarizing its key raised control challenges (see Fig. 1).

We split a WMN into a different number of logical networks, each one configured to autonomously meet distinct levels of users' preferences, and requirements of their services and devices - *users' context*. Users can then be grouped according to similarity on their context, and can be assigned to logical networks matching their context. The logical networks are built through network virtualization, which enables high isolation among communications supported by distinct logical networks (or VNs), and endows our approach with enough flexibility and programmability to control and manage such networks.

2.1 Control and Management Goals

The provision of personalized access for users, which are constantly changing their locations and context, imposes the definition of an intelligent control mechanism to dynamically create, discover, select, extend, or remove the VNs.

First, users' demands need to be sensed and quantified in specific levels or policies, and rules need to be designed to map them into proper VNs' features. Then, and since the creation of VNs to exactly meet the demands of a single user cannot be always performed due to its complexity or unavailability of wireless

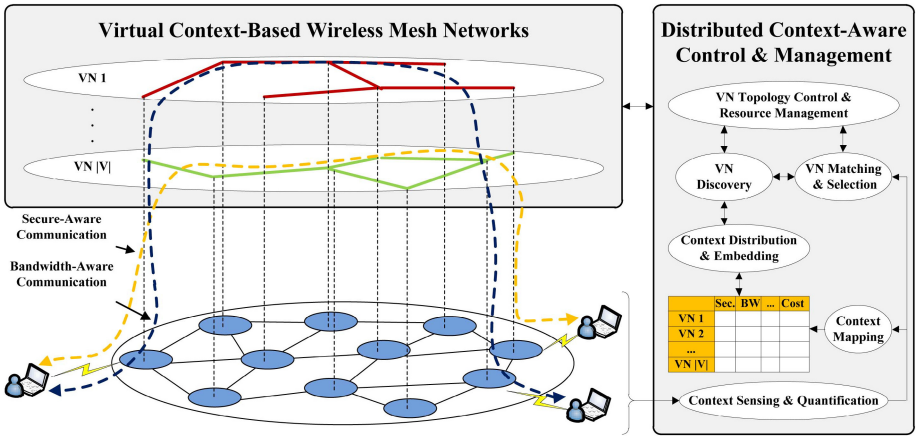

Fig. 1. Concept Overview and Challenges

resources, we will take into consideration the flexibility of users to be attached to VNs that, in the one hand, do not exactly fit all their demands, but on the other hand, meet several allowed variation ranges of such demands. Further, context-aware similarity metrics need to be defined to match users' demands against VNs' features, being selected the best fitting available VNs for users.

The high dynamics of WMN environments fosters the need to constantly discover fitting VNs for users. In order to leave behind the drawbacks of centralized solutions, we define a structured-based topology-aware distributed framework to perform context-driven discovery of VNs on-demand. In our architecture, the WMN nodes (and their supported VN nodes) are endowed with enhanced autonomous capabilities to: *(i)* co-operatively create, maintain, or update the aforementioned context-aware distributed control framework; *(ii)* take advantage of information about the availability of WMN resources and the perceived QoE of ongoing users to limit the candidate set of fitting VNs for users.

Although this paper already proposes a mechanism to select VNs and to control their topologies based on resource- and QoE-aware metrics (normalized through specific fuzzy-based logic functions), we consider the availability of a learning-based scheme to dynamically monitor and update the VN flows' paths based on the perceived QoE of VN users. In such scheme, the perceived QoE of VN users will be derived from the VN context purpose, and from the end-to-end QoS parameters of users' communications conveyed in data packet headers or data ACK messages. We also consider the availability of a mechanism to dynamically compute and update the levels of wireless resources that need to be assigned to the nodes that are part of a particular VN based on the context purpose and number of flows of such VN. These levels will then be the input to an overall distributed mechanism to dynamically map, schedule, and switch WMN interfaces and channels to optimally serve the multitude of VNs.

2.2 Architectural Model Preliminaries

The WMN is formally defined as (N, L), where N=$\{n_1, ..., n_{|N|}\}$ represents the set of WMN nodes (in the overall paper, $|X|$ is the number of elements of the set X). L is the set of WMN links between neighboring elements of N, being defined as L=$\{(n_a, n_b) \in N \mid n_a$ is in transmission range of $n_b\}$. The shortest path for a communication performed between n_w,$n_z$$\in N$ is defined as $L_{n_w \to n_z}$.

Our architecture is driven by a set C=$\{c_1, ..., c_{|C|}\}$ of users' context demands. Each $c \in C$ may be quantified into M_c=$\{1, 2, 3, ..., |M_c|\}$ normalized levels.

A set U=$\{u_1, ..., u_{|U|}\}$ of users may access the WMN, and the attached WMN node of each $u \in U$ is $n_u \in N$. The context demands of each $u \in U$ and their maximum allowed variation ranges are represented by R_u=$\{r_{u_c} \mid r_{u_c} \in M_c$, $c \in C\}$ and T_u=$\{t_{u_c} \mid t_{u_c} \in M_c$, $c \in C\}$, respectively.

The WMN will be the substrate for a set V of possible VNs. Each $v \in V$ is properly configured to meet a set of $R_v = \{r_{v_c} \mid r_{v_c} \in M_c, c \in C\}$ context levels. The identifier of each $v \in V$ is related with R_v; e.g., if $|C| = 4$, and $|M_c| = 5$ $(\forall c \in C)$, the $v_{ID} = 3333$ is representative of a $v \in V$ that is configured to meet the normalized level 3 on each $c \in C$. Each $v \in V$ makes use of a set $N_v \subset N$ of WMN nodes. Each $n \in N$ is the substrate for nodes of a set $V_n \subset V$ of VNs.

The $v'_u \in V$ exactly fits the demands of $u \in U$. The set V'_u=$\{v \in V | \forall c \in C, |r_{u_c} - r_{v_c}| \leq t_{u_c}$ && $\exists c \in C, |r_{u_c} - r_{v_c}| \neq 0\}$ is composed by VNs that do not exactly fit the demands of u, but each feature of $v \in V'_u$ meets its respective allowed variation range (partially fitting VNs for u). The associated VN to $u \in U$ is v_u, being selected from the candidate set V_u=$v'_u \cup V'_u$. The remaining set $\neg V_u$=$\{v \in V | \exists c \in C, |r_{u_c} - r_{v_c}| > t_{u_c}\}$ is the set of non-fitting VNs for u.

3 Context Mapping, Variation, and VN Selection

Although there may be a large set of context parameters, this section considers a small set with demonstrative context information. We consider the set

$$C_1 = \{Cost, Security, Energy, Service\ Type\}.$$

However, mobility, communication type, or preferred access technology are examples of other parameters. This section presents rules to map each $c \in C_1$ into proper VNs' features (summarized in Table 1). Then, we assess at the impact of the variation of each $c \in C_1$ in the architectural functionality, and propose a metric to select the best fitting VN for a user from the set of candidate ones.

3.1 Context Mapping

This sub-section presents several rules to map users' demands in VNs' features. Notice that these are examples, and other rules could be applied.

Cost. From a user perspective, cost preferences may be related to the less or more money that users are willing to pay to have access to low or high reliable communications, respectively. From a network perspective, WMN providers may

Table 1. Context-Aware Mapping of Users' Demands in VNs' Features

User Demand (R_u)	VN Feature (R_v)
Cost Preferences	• Revenues from Multi-Path Availability
Security Constraints	• Authentication, Authorization, and Accounting • Encryption and IPSec-Aware Mechanisms
Energy Consumption	• Interference-, and Congestion-Aware Protocols • Average Node Degree and Number of Links
Service Type	• Resource Management, Routing, and Path Control

lease their infrastructure to VN providers at a different cost, and such cost refers to the total number of required VN nodes to provide a specific level of VN multi-path redundancy (or VN reliability). So, distinct users' cost preferences will be directly mapped into distinct levels of VN multi-path redundancy.

Security. Distinct security and trust constraints of users will determine the authentication, accounting, and authorization protocols running in a VN, as well as the encryption mechanisms and IPSec-aware protocols of such VN.

Energy. Users' devices with strict energy requirements will certainly benefit from associations to VNs that provide the necessary connectivity, while still minimizing energy wasting and maximizing the resource usage. So, high energy-efficient VNs need to reduce the: *(i)* number of collisions to not waste energy in packet retransmissions; *(ii)* number of traversed hops of VN communications by minimizing the average VN node degree and the number of VN links.

Service Type. Quality requirements of users' services may lead to a preliminary grouping of users according to similarity on these requirements. Then, these requirements will also play an important role in the subsequent VN routing and resource management mechanisms, such as: *(i)* the selection of the nodes that need to be added to extend or create a VN; *(ii)* the mapping, scheduling, and switching of VN links to WMN interfaces and channels; *(iii)* the configuration of the buffer size, processing power, and memory of VN nodes.

3.2 Context Variation

The demands of each $u \in U$ are characterized by distinct levels of flexibility to enable the association of u to a VN that does not exactly fit them. Such flexibility (or allowed variation range during the connectivity time of u, T_u) has to be efficiently embedded on the: *(i)* selection of the best fitting VN for u (detailed in the next sub-section); *(ii)* organization of the features and levels that characterize the VNs to enable optimized distributed discovery and adaptation of VNs (detailed in the next two sections). This sub-section provides some qualitative insights on the allowed flexibility of several context features.

First, VN providers aim to increase their revenues from providing multi-path redundancy, and in a contradicting perspective, users are usually reluctant to exceed their cost preferences. Therefore, the accepted variation range of cost preferences during users' connectivity times is extremely restrictive.

Since security constraints are related to services' purposes and trustability, variations of these demands are also rarely allowed from the user perspective.

On the other hand, the quality requirements of users' services or the energy requirements of users' devices may admit high variation ranges, since users may very often change their required services, or may easily have access to devices' battery charging mechanisms. So, these two context features have a less important role in the preliminary context-aware VN selection process for users, since there are other restrictive context parameters that need to be exactly met during users' connectivity times. However, they will have a high impact on the network-centric VN path selection and resource management mechanisms.

3.3 Context-Aware VN Matching and Selection

Due to the possibility of $u \in U$ to be connected in more than one $v \in V$, this sub-section defines a metric to derive the context-aware **S**imilarity **D**ifference (SD) among the demands of u (R_u) and the features of each $v \in V$ (R_v), and to select the $v \in V$ with the highest probability to deal with variations of R_u.

In the one hand, it is easily inferred that if v belongs to a non-fitting VN for u, $v \in \neg V_u \Rightarrow SD_{(u,v)}=\infty$. On the other hand, if $v \in V_u$, and due to the multitude of elements of C and M_c, $SD_{(u,v)}$ will be a weighted sum of the absolute single differences among r_{u_c} and r_{v_c} ($\forall c \in C$). The weight associated to the single absolute difference on each $c \in C$ is inversely proportional to the allowed variation range t_{u_c}, that is, the relative importance of $|r_{u_c} - r_{v_c}|$ increases with a lower t_{u_c} (where t_{u_c} represents the maximum allowed variation of r_{u_c}). Following this way, we will clearly reduce the number of future required updates on the selected $v_u \in V_u$ due to variations of R_u. So, our metric is formally defined as follows:

$$SD_{(u,v)}(R_u, T_u, R_v) = \begin{cases} \sum_{i=1}^{|C|} \frac{|r_{u_{c_i}} - r_{v_{c_i}}|}{t_{u_{c_i}}} & , v \in V_u \\ \infty & , v \in \neg V_u \end{cases} . \quad (1)$$

As an example of application of our metric in the VN selection process, consider that $|C|=4$, $|M_c| = 5$ ($\forall c \in C$), $R_u=\{3,3,3,3\}$, $T_u=\{1,2,2,2\}$, and there are two fitting VNs for u: $v_1, v_2 \in V_u$, where $R_{v_1}=\{3,2,3,3\}$ and $R_{v_2}=\{2,3,3,3\}$, respectively. Although $\sum |R_u - R_{v_1}| = \sum |R_u - R_{v_2}|$, the impact of a future variation of $r_{u_{c_1}}$ is higher than if such change occurs on $r_{u_{c_2}}$, since $t_{u_{c_1}} < t_{u_{c_2}}$. So, v_1 will be selected as v_u, since $r_{u_{c_1}} = r_{v_{1_{c_1}}}$, while v_2 is already in the lower border to meet $r_{u_{c_1}}$. Both v_1 and v_2, if selected, still have a high probability to allow a future variation of $r_{u_{c_2}}$, and so, c_2 has associated a lower weight than c_1.

Finally, the selected VN for u, v_u, will be:

$$v_u \in V_u \quad s.t. \quad SD_{(u,v_u)} = \min_{v \in V_u} \{SD_{(u,v)}\}. \quad (2)$$

4 Distributed Context-Aware Control Framework

This section defines the basis of a context-aware distributed framework to control the multitude of VNs. Details on the elements that compose such framework and

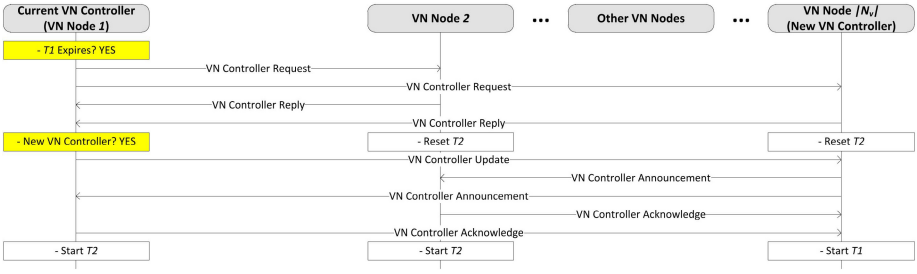

Fig. 2. VN Controller Update

on the mechanisms to inter-connect them are also provided. In the next section, we will make use of the proposed framework to discover fitting VNs for users on-demand, and to control and manage VN topologies in a distributed way.

4.1 VN Controller

The basic element of our distributed control framework is the VN controller ($O_v \in N_v$, $v \in V$). A VN controller is a VN node that stores the relevant VN information (e.g., context demands, and perceived QoE of ongoing VN users) that is used to perform intra-VN control functionalities (e.g., VN topology, path, and resource control), and to allow an optimized distributed selection of a fitting VN for a user in the WMN (detailed in the next section).

At the time of the creation of $v \in V$, it is randomly chosen a node of v to be the O_v, which then needs to be updated to be adapted to topology changes of v. According to Fig. 2, O_v periodically triggers a process to refresh the stored control information of v (the timer T_1 is related to the periodicity of such process). Then, each $n_v \in N_v$ informs O_v about the requested control information, and resets its timer T_2 (used to detect a misbehavior of O_v, as will be explained below). In the following, and since we aim to minimize the physical topology distance between every $n_v \in N_v$ and O_v to enable small time consuming intra-VN control communications, the new O_v is designated as:

$$n_{v_i} \in N_v \quad s.t. \quad \sum_{n_{v_j} \in N_v} L_{n_{v_i} \to n_{v_j}} = \min_{n_{v_k} \in N_v} \{ \sum_{n_{v_j} \in N_v} L_{n_{v_k} \to n_{v_j}} \}. \tag{3}$$

Notice that the incorporation of other metrics to select O_v, such as the resource availability and stability of each $n_v \in N_v$, can be addressed in the future.

If another VN node is selected to be O_v, it is notified by the current O_v, and then, the new O_v announces itself in the VN. If $n_v \in N_v$ did not receive any message from O_v in the time-out T_2, a new O_v has to be selected. In order to easily select another O_v in a synchronized way, O_v periodically provides an ordered list of the nodes of v according to their rankings to be designated as O_v (obtained through (3)). Then, each $n_v \in N_v$ configures its timer T_2 according to its position p_{n_v} in such list ($T_{2_{n_v}}$=$T_2 \times p_{n_v}$). When a problem occurs with O_v,

the first node of the list provided by O_v is able to auto-designate and announce itself as the new O_v. Through this approach, we can even deal with partitions in VNs, allowing each VN partition to autonomously select its VN controller. Notice that both T_1 and T_2 are dependent from the context purpose of each VN.

4.2 Distributed Control Structure of VN Controllers

This sub-section proposes a control ring to inter-connect the available VN controllers based on Distributed Hash Tables (DHTs) (see Fig. 3). The multi-context and multi-level features that characterize the VNs and the users' allowed context flexibility are embedded in the DHTs' characterization and organization in order to efficiently allow the VN controllers that often need to communicate to each other to be closely located in the ring. Moreover, this section details the processes to maintain the DHT-based routing entries stored by each VN controller.

4.2.1 Context Space Partition and Organization

By assessing the level of flexibility (or allowed variation range, t_{u_c}) of each $c \in C$ (please refer to sub-section 3.2), we first split C in two disjoint sets $C = C' \cup \neg C'$, where C' is composed by $c \in C$ admitting low values of t_{u_c}, and $\neg C'$ is composed by $c \in C$ admitting high values of t_{u_c}. Here, C' contains the majority of the features that do not frequently admit variations during users' connectivity times (e.g., cost or security), and almost never trigger adaptation of users' associated VNs; on the other hand, $\neg C'$ contains the remaining features that, in general, admit large variations during users' connectivity times (e.g., service type), and constantly trigger adaptations of users' associated VNs.

This distinction in the flexibility of context levels is very important, since we can have a first notion of the context parameters that have more probability to change during users' connectivity times. Then, a ring structure that enables small time consuming communications among the controllers of VNs that are most probable to fit the users' context demands and their variations, allow to quickly discover and adapt fitting VNs for users based on the co-operation among the different elements of the ring.

Following this idea, we split the ring in a set of semantic clusters (SCs), each one grouping the controllers of $\forall v, v' \in V : R_v[|C'|] = R_{v'}[|C'|]$. So, the controllers of VNs that are more probable to fit the most frequent users' context changes belong to the same SC, being such controllers closely located in the ring.

Within each SC, the $c \in \neg C'$ with the highest t_{u_c} has the most important role on the context-aware proximity among VN controllers, since it is the context parameter with the highest probability to change. So, the controllers of two VNs that only differ in one level in the $c \in \neg C'$ with the highest t_{u_c} are 1-hop logical neighbors in the ring, storing a direct DHT-based routing entry to communicate to each other.

Fig. 3 presents an example of the ring structure by considering $C = \{c_1, c_2, c_3\}$, $C' = \{c_1, c_2\}$, $\neg C' = \{c_3\}$, $|M_{c_1}| = 2$, $|M_{c_2}| = 4$, and $|M_{c_3}| = 4$. First, the total number of possible available SCs is given by $|M_{c_1}||M_{c_2}| = 8$; e.g., SC_3 groups the controllers of available VNs characterized by the level 1 on $c_1 \in C'$, and the

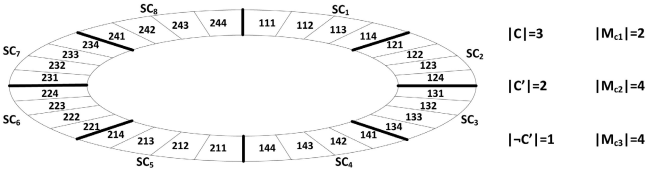

Fig. 3. Context-Aware Ring-Based Control Structure of VN Controllers

level 3 on $c_2 \in C'$. Second, the SCs and the VN controllers that belong to the same SC are organized in a consecutive order of the levels of the VN features that are part of C' and $\neg C'$, respectively. Third, c_3 has associated the highest t_{u_c}, and so, the controllers of two VNs that only differ in one c_3 level in the context-aware characterization are 1-hop logical neighbors in the ring.

4.2.2 Logical Control Links

In the following, we detail the control processes that allow: *(i)* WMN nodes to store a set of entries in their routing tables to access to the several SCs of the ring; *(ii)* VN controllers to store a set of entries to perform intra-SC routing. Both DHT-based routing entries will be used to accelerate the distributed discoveries of fitting VNs for users, as will be detailed in the next section.

Disseminated Control Information. In our approach, each $n \in N$ periodically announces its local VN nodes to its 1-hop neighbors (the timer T_3 is the periodicity of such process). Such information is then disseminated by the neighbors of n along a pre-defined maximum number of hops TTL_{max}. The information tuple associated with each announced node of $v \in V$ is defined by:

$$t_v = \{n_{ID}, v_{ID}, TTL\}, \tag{4}$$

where, n_{ID} is the identifier of the WMN node that provides support for the node of v, v_{ID} is the identifier of v, and $TTL \leq TTL_{max}$ indicates the maximum amount of hops that this information tuple can be forwarded in the WMN.

Links to SCs. To limit the amount of control information disseminated in the WMN, each $n \in N$ only broadcasts to its neighbors one information tuple to access to each SC of the control ring. For instance, and from the set T_v of local or gathered information tuples stored by n, n will select to broadcast to its neighbors the information tuple to access to SC_1, t_{SC_1}, according to:

$$t_{SC_1} = \{\forall t_v \in T_v \mid t_v.TTL = \max_{\{v|O_v \in SC_1\}} \{t_v.TTL\}\}. \tag{5}$$

From (5), our approach both minimizes the length of the routes to WMN nodes get access to the several SCs, and balances the control responsibilities among the different elements of each SC (due to the random selection).

Intra-SCs Links. In order to endow the distributed control framework with enough flexibility to deal with the dynamics of WMN environments, we will

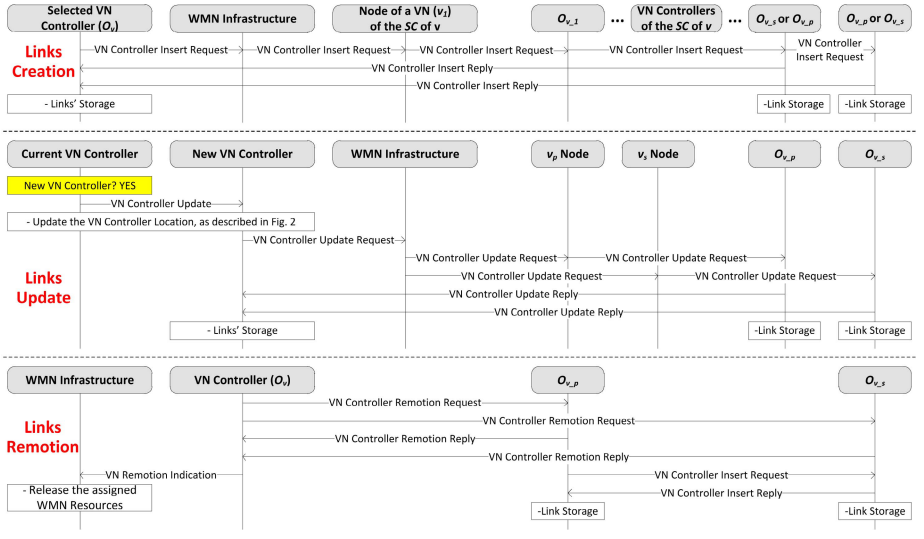

Fig. 4. Intra-SC Links Control and Management

now detail the control processes to create, update, and remove the DHT-based routing entries among VN controllers belonging to the same SC (depicted in Fig. 4).

A) Links Creation. In our approach, each O_v stores a routing entry to access to O_{v_s} (the controller of the immediately successor VN of v, v_s) and another one to access to O_{v_p} (the controller of the immediately predecessor VN of v, v_p). O_{v_s} and O_{v_p} are 1-hop logical neighbors of O_v in the control ring.

According to the top part of Fig. 4, at the time of the creation of v, the selected O_v has to be integrated in the control ring through the establishment of these two routing paths. In such process, O_v first contacts its substrate WMN node, which, based on its stored routing entries, contacts the nearest neighbor that provides support for a VN with controller belonging to the SC of O_v (e.g., v_1). After being contacted by one v_1 node, O_{v_1} contacts its 1-hop logical neighbor that is closely located to O_{v_s} and to O_{v_p} in such SC. Such process is recursively performed along the SC in a DHT-alike way until finding O_{v_s} and O_{v_p}. Finally, O_v is notified and stores the routes to contact O_{v_s} and O_{v_p}.

B) Links Update. In the one hand, O_v has to be closely located to every $n_v \in N_v$ to reduce the delays of intra-VN control communications (please refer to (3)). On the other hand, and to avoid topology mismatching problems between the WMN infrastructure and the control ring, the physical topology distance from O_v to O_{v_s} or to O_{v_p} needs to be minimized to enable small time consuming communications in the ring. This last point fosters the need to update the O_v selection function proposed in (3), which is detailed below.

According to Fig. 2, when O_v starts a process to refresh the stored control information of v, it also announces the IDs of v_s and v_p. Then, each $n_v \in N_v$ asks

its substrate WMN node about the existence of nodes of v_s or v_p in its physical neighborhood. Based on its stored routing entries, such WMN node informs n_v about the nearest location of nodes of v_s or v_p, which is also encapsulated in the reply sent to O_v. Then, the new O_v is selected according to:

$$n_{v_i} \in N_v \quad s.t. \quad L_{n_{v_i}} + L'_{n_{v_i}} = \min_{n_{v_j} \in N_v} \{L_{n_{v_j}} + L'_{n_{v_j}}\}, \tag{6}$$

$$L_{n_{v_j}} = \sum_{n_{v_k} \in N_v} L_{n_{v_j} \rightarrow n_{v_k}} \quad \text{and} \quad L'_{n_{v_j}} = 0.5 \min_{n_{v_{s_k}} \in N_{v_s}} \{L_{n_{v_j} \rightarrow n_{v_{s_k}}}\} + 0.5 \min_{n_{v_{p_k}} \in N_{v_p}} \{L_{n_{v_j} \rightarrow n_{v_{p_k}}}\}. \tag{7}$$

According to (6) and (7), the selected O_v is the $n_{v_i} \in N_v$ that both minimizes the distance to every other node of v ($L_{n_{v_i}}$), and the mean distance to access to the successor and predecessor VNs of v in the control ring ($L'_{n_{v_i}}$).

If another node of v is selected to be O_v, the process depicted in Fig. 2 is started in v. Moreover, and in order to update the routes between O_v and O_{v_s} or O_{v_p}, the process depicted in the middle part of Fig. 4 is started in the WMN. In such process, O_v first contacts its substrate WMN node, which then notifies the neighbors that provide support for the selected nodes of v_s and v_p. Then, O_{v_s} and O_{v_p} are notified about the routes to contact O_v, and then reply to O_v.

C) Links Remotion. According to the bottom part of Fig. 4, if O_v has to leave the WMN due to the complete remotion of v (the mechanism to decide when to remove VNs will not be detailed in this paper due to the space limitations), it will notify such intention to O_{v_s} and to O_{v_p} to adapt the control ring (such adaptation can be also triggered by O_{v_s} or O_{v_p}, if they did not receive any routing maintenance message from O_v in the time-out T_4). These notification messages will convey the information to allow O_{v_s} and O_{v_p} to become 1-hop logical neighbors in the ring, and they can then announce themselves to each other.

4.3 Global VN Manager

Finally, our proposed control framework is composed by an entity, the global VN manager, which stores the IDs of the available VNs in the WMN, and a link to access to one of their nodes. By storing this minimum knowledge, the global VN manager is able to help the proposed control processes.

For instance, and in case of a small number or high sparseness of VNs in the WMN, a specific WMN node may not store any route to contact SC_1, where a new VN controller has to be integrated. In such situation, the time-out to create the new links in the ring reaches a critical value, and the global VN manager is notified to quickly redirect the process to SC_1.

5 User Connectivity and VN Topology Control

This section makes use of the presented distributed framework to propose a control mechanism to associate users to fitting VNs and to manage VN topologies

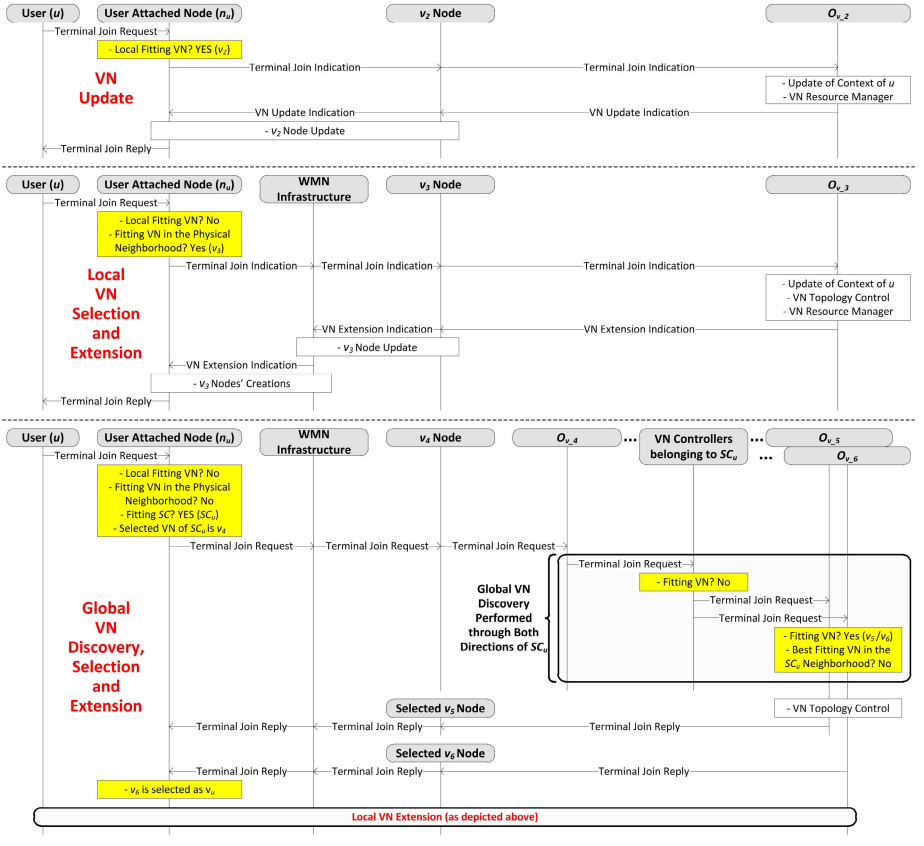

Fig. 5. User Connectivity and VN Update, or Local/Global VN Extension

on-demand. From the point of view of a user that arrives at a specific WMN node and wants to be connected in a fitting VN (or even changes his/her context requirements), this section presents the required network control processes to enable the: *(i)* selection and update of the best fitting VN from the set of candidate ones available in the user attached WMN node; *(ii)* local or global (context-aware driven) discovery of a fitting VN for the user; *(iii)* selection of the best path to extend a fitting VN to be adapted to the user's location. The control processes and the used criteria to decide when to create a new VN or to remove unused VNs will be left for future work due to space limitations.

5.1 VN Update

According to the top part of Fig.5, when $u \in U$ arrives at $n_u \in N$, u sends its context demands to n_u. Then, n_u quantifies these demands in levels (R_u, T_u), and matches them against the features of each $v \in V_{n_u}$ (R_v). If there are any $v \in V_u$ in n_u, and v_2 is selected according to (2) to be v_u, n_u contacts its local v_2 node

to notify O_{v_2}. In the following, and after storing the context of u, O_{v_2} runs the VN resource management algorithm to update the assigned resources to the v_2 node supported by n_u (as described in Section 2, the distributed management of WMN resources to serve the multitude of VNs is out of the scope of this paper). Finally, u is notified about the establishment of his/her connection to v_2.

5.2 Local VN Selection and Extension

Focusing now in the middle part of Fig.5, if n_u does not support any $v \in V_u$, n_u will inspect the gathered VN information from its neighborhood. If n_u has information of more than one $v \in V_u$ available in its neighborhood, n_u has to select the best fitting VN node to trigger a local VN extension up to n_u.

In our approach, this selection process will be aware of the resource availability in the path between n_u and the WMN node supporting a specific fitting VN node for u, and of QoE indicators of satisfaction of ongoing users in such VN node. For this purpose, the information tuple of each node of $v \in V$ announced by $n \in N$ to its neighbors (please refer to (4)) has to include two more fields:

$$t_v = \{n_{ID}, v_{ID}, TTL, f(R_{L_{n_v \to n_u}}), f'(QoE_v)\}, \tag{8}$$

where f and f' are two fuzzy-based logic functions used to respectively map in Z normalized levels the: *(i)* overall WMN resource availability in the path between the WMN node supporting the v network and the u attached node, $R_{L_{n_v \to n_u}}$; *(ii)* mean perceived QoE of ongoing users in the v network, QoE_v. For instance, if $Z = 3$, we will have:

$$f(R_{L_{n_v \to n_u}}) = \begin{cases} 1 & , \ R_{L_{n_v \to n_u}} \leq Z_1 \\ 2 & , \ Z_1 < R_{L_{n_v \to n_u}} \leq Z_2 \\ 3 & , \ Z_2 < R_{L_{n_v \to n_u}} \end{cases}, \ f'(QoE_v) = \begin{cases} 1 & , \ QoE_v \leq Z'_1 \\ 2 & , \ Z'_1 < QoE_v \leq Z'_2 \\ 3 & , \ Z'_2 < QoE_v \end{cases}. \tag{9}$$

From the set T_v of gathered information tuples stored by n_u, n_u selects the one of a $v \in V_u$, t_u, according to:

$$t_u = \{t_v \in T_v | \ t_v.TTL = \max_{v \in V_u}\{t_v.TTL\} \ \&\& \\ t_v.f(R_{L_{n_v \to n_u}}) + t_v.f'(QoE_v) = \max_{v \in V_u}\{t_v.f(R_{L_{n_v \to n_u}}) + t_v.f'(QoE_v)\}\}. \tag{10}$$

From (10), n_u first limits the candidate set of fitting VN nodes for u to the ones accessible in a minimum number of hops, since they enable the creation of a less number of virtual nodes in a future VN extension, which is a high resource and time consuming process. Then, and from the remaining information tuples, n_u gives preference to the ones allowing VN extensions through non-congested WMN paths, and users' associations to VNs with high QoE indicators.

Supposing that v_3 is selected to be v_u, n_u contacts the WMN node that provides support for the selected v_3 node to notify O_{v_3}. Then, and after storing the context of u, O_{v_3} updates the VN topology information, and runs the VN resource management algorithm to derive the resources that need to be assigned

to the new v_3 nodes. In the following, new v_3 nodes are created in the WMN nodes where the v_3 extension takes place. Finally, u is connected to v_3.

5.3 Global VN Discovery, Selection and Extension

If n_u does not support and does not have information of any $v \in V_u$, n_u will trigger a global discovery process in the WMN using the distributed control framework. Such discovery will be followed by the extension of the $v \in V_u$ up to n_u. This two-step control process is depicted in the bottom part of Fig.5.

5.3.1 Global VN Discovery

The global context-driven discovery process of any $v \in V_u$ in the WMN has two major steps: *(i)* the redirection of the process to a proper SC of the control ring; *(ii)* the discovery performed within the selected SC.

Redirecting the Discovery to a Fitting SC. Based on the routes stored by n_u to access to the several SCs, which are represented through the set T_v of gathered information tuples, n_u first makes use of a random tuple that allows to access to the SC that is semantically closer to the first $|C|$ demands of u, SC_u, in a minimum number of hops. Such information tuple, t_{SC_u}, is defined by:

$$t_{SC_u}=\{\forall t_v \in T_v \mid SD_{(u,v)}[|C|]=\min_{v\in V_u}\{SD_{(u,v)}[|C|]\} \&\& t_v.TTL=\max_{v\in V_u}\{t_v.TTL\}\}. \tag{11}$$

Considering that v_4 is the selected VN with controller belonging to SC_u, n_u then contacts the WMN node supporting the selected v_4 node to notify O_{v_4}.

Discovery within the Selected SC. In the following, O_{v_4} forwards the global VN discovery process through both clockwise and counterclockwise directions of SC_u by using the routes to contact its 1-hop logical neighbors (as explained in the previous section). Such process is recursively performed along SC_u in a DHT-alike way until finding the controller of any $v \in V_u$.

Focusing in a specific SC_u direction, after finding the controller of any $v \in V_u$, such controller may be followed by a controller of another $v' \in V_u$ due to the context-aware organization of the control ring. In such situation, O_v may take advantage of specific quality information of v' to optimize the VN discovery at the cost of one more communication in the control ring. By conveying indicators of the mean QoE of the ongoing VN users in the control messages used to periodically update the links among consecutive VN controllers (please refer to the middle part of Fig. 4), O_v will inspect such information. Then, O_v will only redirect the global VN discovery process to $O_{v'}$ using its stored DHT-based routing entry, if v' is characterized by a higher level of mean perceived QoE of its users than v.

If the controller of any $v \in V_u$ in SC_u is not available, the WMN nodes that provide support for the VN controllers of SC_u, in which the global VN discovery process stops, are notified to redirect the discovery to the following SC selected according to (11). This two-step discovery process is recursively performed in the control ring until finding the controller of any $v \in V_u$ in any SC.

5.3.2 Global VN Selection and Extension

Considering that v_5 and v_6 are the fitting VNs for u that are discovered and selected in the clockwise and counterclockwise directions of SC_u, respectively, O_{v_5} and O_{v_6} will then notify the near v_5 and v_6 nodes to n_u about the possibility of their extensions up to n_u. In the following, the WMN nodes that provide support for the selected v_5 and v_6 nodes notify n_u.

From the obtained replies, which should contain information about $R_{L_{n_v \to n_u}}$ and QoE_v, n_u selects the $v \in V_u$ to be extended up to n_u according to (10). Supposing that v_6 is selected to be v_u, a VN extension process similar to the one explained in the previous sub-section is started in the WMN.

6 Conclusion and Future Work

This paper considers the support of context-based WMNs through their splitting in several VNs, each one created and configured to meet a specific set of users' context preferences and requirements of their services. We propose a distributed and adaptable DHT-based framework to associate users to fitting VNs and to control and manage VNs on-demand based on context-, topology-, resource-, and QoE-aware metrics. The elements of such framework and their inter-connections are characterized and organized to efficiently deal with the set of considered context features.

However, there are several aspects that need future research, such as, the distributed resource management mechanism, the definition of the QoE-aware intra-VN routing scheme, and the concretization of the fuzzy-based logic functions to map resource and QoE metrics in normalized levels. With respect to evaluation, we plan to assess the impact of the sizes of the sets C, M_c ($\forall c \in C$), and C' and the value of the parameter TTL_{max} in the amount of control information disseminated in the WMN to maintain the multitude of SCs, and in the number of communications required to discover a VN. Finally, we plan to compare the efficiency and impact of the proposed control approach against other centralized or distributed solutions available in the literature.

Acknowledgments. This research was supported in part by Fundação para a Ciência e Tecnologia, grant SFRH/BD/45508/2008.

References

1. Madwifi, `http://madwifi-project.org/`
2. Akyildiz, I., Wang, X.: A Survey on Wireless Mesh Networks. IEEE Communications Magazine 43(9) (September 2005)
3. Ding, G., et al.: Overlays on Wireless Mesh Networks: Implementation and Cross-Layer Searching. In: IFIP/IEEE MMNS (2006)
4. Matos, R., et al.: Analytical Modeling of Context-Based Multi-Virtual Wireless Mesh Networks. Special Issue on "Models and Algorithms for Wireless Mesh Networks" in the "Ad Hoc Networks" Elsevier Journal (accepted, 2011)

5. Matos, R., et al.: Context-based Wireless Mesh Networks: A Case for Network Virtualization. Special Issue on "Future Internet Services and Architectures: Trends and Visions" in the "Telecommunication Systems" Springer Journal 52(3) (2013)
6. Anderson, T., et al.: Overcoming the Internet Impasse through Virtualization. IEEE Computer 38(4) (April 2005)
7. Dey, A.K.: Providing Architectural Support for Building Context-Aware Applications. Ph.D. thesis, Atlanta, GA, USA (2000)
8. Giustiniano, D., Goma, E., Lopez, A., Rodriguez, P.: WiSwitcher: An Efficient Client for Managing Multiple APs. In: ACM PRESTO (2009)
9. Hu, P., Portmann, M., Robinson, R., Indulska, J.: Context-Aware Routing in Wireless Mesh Networks. In: ACM CASEMANS (2008)
10. Oh, M.: An Adaptive Routing Algorithm for Wireless Mesh Networks. In: ICACT (February 2008)
11. Shrestha, S., Lee, J., Chong, S.: Virtualization and Slicing of Wireless Mesh Network. In: Conference on Future Internet (CFI) (June 2008)