A Survey of Advanced Ethernet Forwarding Approaches

Rute C. Sofia

***Abstract*—The higher transmission rates currently supported by Ethernet lead to the possibility of expanding Ethernet beyondthe *Local Area Network* scope, bringing it into the core of large scale networks, of which a *Metropolitan Area Network (MAN)* is a significant example. However, originally Ethernet was not devised to scale in such environments: its design does not contemplate essential requirements of larger and more complex networks, such as the need for resilience, scalability, or even integrated control features. Furthermore, its spanning-tree based forwarding results in slow convergence and weak resource efficiency.**

Specifically focusing on Ethernet's forwarding behaviour, this survey covers solutions that enhance the Ethernet's path computation, allowing it to scale in larger, more complex environments. General notions concerning the application of Ethernet in Metro areas are also provided, as a specific example of Ethernet's application in large scale networks.

***Index Terms*—Carrier-grade Ethernet, MAN, forwarding.**

I. Introduction

THE RECENT advances introduced in Ethernet technology (such as the higher transmission rates) lead to the possibility of deploying Ethernet within the core of large scale networks, of which a *Metropolitan Area Network (MAN)* is a relevant example. Ethernet's connectionless nature is adequate to the support of IP-based services, and its flexibility allows the deployment of novel types of infrastructures, e.g., multipoint-to-multipoint services, which can provide better bandwidth efficiency and which require less global state information, when compared to other, connection-oriented transport solutions.

While promising, the original scope of Ethernet was limited to *Local Area Networks (LANs)*. Consequently, its design falls short in terms of MAN requirements such as resilience, scalability, or even integrated control features [1]. Furthermore, on its original format, Ethernet relies on a spanning-tree aproach (Spanning Tree Protocol (STP)/Rapid Spanning Tree (RSTP) [2]) to perform forwarding. STP gives the means to provide a simple but non-optimal forwarding, by performing *loop avoidance*. STP creates a logical topology in the form of a spanning-tree where the path from every node to the root bridge is a shortest-path in the form of a min-cost (cumulative link cost) path. The choice of the bridge that plays the role of root therefore strongly dictates the efficiency of the resulting logical topology. Hence, there is no guarantee that the path between any two nodes is a shortest-path. In a MAN, not only does STP converge slowly but it also prevents the use of some links, given that it avoids loops by means of relying on tree topologies. And, for the case of ring topologies, STP application requires the support of protocols such as *Ethernet Ring Protection (ERP)* [3] or *Ethernet Automatic Protection Switching (EAPS)* [4] to provide better reliability and resilience: while RSTP provides basic resilience in rings, EAPS or ERP can provide faster failover recovery in rings.

The realization of the mentioned drawbacks lead to the appearance of STP enhanced standards such as the *RSTP* [5] (now incorporated into [2]) or the *Multiple Spanning Tree Protocol (MSTP) [6]*, which partially solve STP scalability problems. Yet, the resulting end-to-end paths follow the same algorithm and thus resource usage is still not optimized. For instance, it may result in traffic concentration or even traffic losses, when temporary (*transient*) loops occur. Using a spanning-tree is, as mentioned, a simple way to avoid information inconsistency (due to loop avoidance) but quite restrictive particularly when the physical topologies in question are either partially (or fully) meshed, or ring topologies, as is normally the case in MANs.

There are currently several approaches whose main goal is to leverage Ethernet to a carrier-grade stage. In such context, this survey concentrates on work focused on **forwarding enhancement** directions. To better introduce this problem space, section 2 provides terminology, notions, and services being defined by standardization bodies in what concerns Ethernet applied to MANs, i.e., *Metro Ethernet (ME)*, as a significant example of Ethernet's applicability to large scale networks. In section 3 we provide an overview of current IEEE Ethernet standards, namely, STP, RSTP, MSTP. Section 4 gives insight into solutions that provide forwarding enhancements still based on spanning-trees, while section 5 provides an overview of connectionless solutions that are not based on spanning-trees. In section 6, the most popular connection-oriented Ethernet approaches are described. We conclude in section 7.

II. Ethernet In The MAN Core: Metro Ethernet Notions and services

This section gives an overview on *ME* notions and services, as well as on current traffic-engineering solutions that Ethernet relies upon to scale in large and complex environments. We start by introducing a generic MAN model and by providing a basic comparison to *Asynchronous Transfer Mode (ATM)* as another representative example of a MAN core technology. The section then finalizes with a description of Ethernet service definitions being dictated by different standardization

Manuscript received 22 February 2008; revised 16 January 2009.

Rute Sofia is leader of the Internet Architectures and Networking (IAN) area, UTM, INESC Porto. Campus FEUP, R. Dr. Roberto Frias 378, 4200-655 Porto, Portugal (e-mail: rsofia@inescporto.pt).

Digital Object Identifier 10.1109/SURV.2009.090108.

bodies, to then cover solutions being applied to allow Ethernet to scale to the MAN.

As illustrated in Fig. 1, the MAN is typically a network that spans a metropolitan area interconnecting several sites. Historically, telephone companies provided services across MANs which were normally built upon ring topologies supported by *Synchronous Optical Networking (SONET)* [7]/*Synchronous Digital Hierarchy (SDH)* [8], [9]. *SONET/SDH* is based on *Time Division Multiplexing (TDM)*, technology that is by far more suitable for voice than data. But with the rise of the Internet and the expansion of broadband worldwide, the services that are now provided across MANs are both voice and data, and most of them come from the Internet. Consequently, the legacy TDM technologies are not suitable anymore to the rising service needs. Ethernet, on the other hand, is a potential technology to support the transport of *Internet Protocol (IP)* services, providing enough flexibility to transport current and future IP services that may arise.

To give a better perspective of Ethernet's applicability within the MAN, Fig. 1 provides an example of a MAN and its main regions, namely:

- *Customer Premises (CP)*. These relate to residential or enterprise areas, thus fully controlled by the end-user. The *CP* may incorporate end-user devices such as *Personal Computers (PCs)*, *Set Top Boxes (STBs)*. In addition it also contains *Customer Premises Equipment (CPE)*. The CPE term applies to the networking devices, namely, a customer gateway which can be bridged or routed[1], and an additional device (e.g., *Digital Subscriber Line (DSL)* modem) which has a built-in *Network Terminator (NT)*. The customer gateway has, among other features, the role to provide IP connectivity to one or to several *User Equipments (UEs)*.
- **Access network region**. The access network region comprises in fact several networks that provide connectivity and traffic aggregation between end-users and *Service Providers (SPs)*. The access region is operated by one or more *Network Access Providers (NAPs)* and can be further split into *first mile* (*local-loop*) and *aggregation* regions. The former comprises both the physical connection and optional equipment between the CPE and the *Access Node (AN)*, entry point to the access region. The latter comprises the region where first mile traffic is further aggregated, to be delivered to the regional network. The AN represents a point (in most cases, the first) where several circuits coming from different customers are aggregated. The AN performs the required OSI Layer 2 functions, e.g., port isolation, and may incorporate some OSI Layer 3 functionality, e.g., basic IP routing filtering and/or IP session awareness.
- **Regional network region**. This region interconnects the access network to regional broadband networks. The nomenclature for this region is in fact optional, being most of the time access and regional regions addressed as a whole (cf. Fig. 1). When present, the regional network is operated by one or several *Regional Network Providers (RNPs)*. This region (or the access region, when this one is not present) is terminated by the so-called *Edge Nodes (ENs)*, of which a *Broadband Remote Access Server (BRAS)* [10] is a representative example.

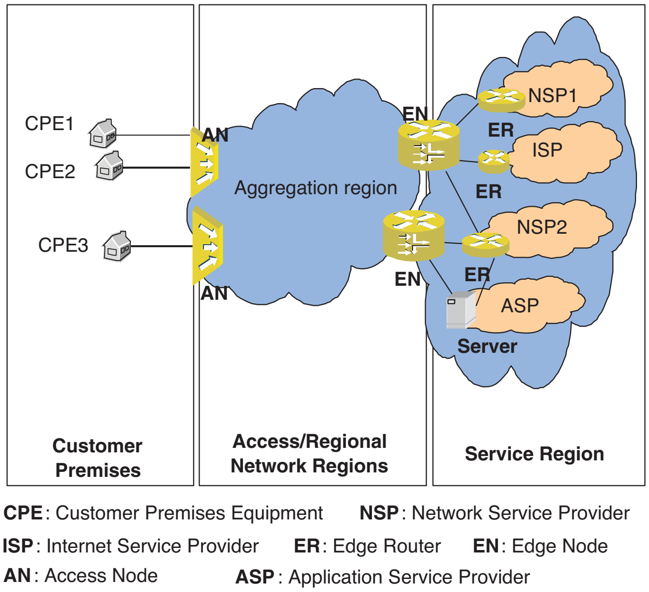

CPE1
CPE2
CPE3
AN
AN
Aggregation region
EN
EN
ER
ER
ER
NSP1
ISP
NSP2
ASP
Server
Customer Premises
Access/Regional Network Regions
Service Region
CPE: Customer Premises Equipment
NSP: Network Service Provider
ISP: Internet Service Provider
ER: Edge Router
EN: Edge Node
AN: Access Node
ASP: Application Service Provider

Fig. 1. MAN reference model.

- **Service Backbone**. This region encompasses networks operated by one or more *Internet Service Provider (ISP)*, *Network Service Provider (NSP)* and *Application Service Provider (ASP)*. This region is therefore in its majority IP-based (*IP/Multi-Protocol Label Switching (MPLS)* [11]) and connects *SPs* to one or more *RNP/NAP*. The *Edge Router (ER)* is the ingress/egress element to/from *ISP/NSP/ASP*, respectively.

The previous notions and model rely on a business perspective to explain the different building blocks of a *MAN*. From a technology point of view and to better explain the concept of *ME*, we rely upon the DSL Forum [12] TR-59 DSL infrastructure model which considers as access/aggregation technologies both ATM [13] or Ethernet [14].

When the MAN core technology used is ATM, then as illustrated in Fig. 2, a *Permanent Virtual Circuit (PVC)* is normally established per end-user (and/or per service), being terminated on the EN which in *DSL*/ATM infrastructures is represented by the *BRAS*. The TR-59 model is therefore a BRAS-centric architecture, where the BRAS holds the required functionality to deal with the aggregated customer traffic. In other words, the *BRAS* represents the aggregation point for traffic coming both from the access/regional networks and from the service region: the BRAS deals with the most varied traffic issues, e.g., *Authentication, Authorisation, Accounting (AAA)*, service differentiation, traffic aggregation, Layer 2/Layer 3 mediation, *Quality of Service (QoS)*, policy enforcement.

The connection to the service region is performed by means of Layer 2 or Layer 3 functionality, i.e., some form of Layer 2 tunneling, IP over bridged Ethernet, or routed IP. If the end-user traffic aggregation is performed at the *Point-to-Point Protocol (PPP)* level, then the received *PPP* traffic has to be split and routed over some form of Layer 2 tunneling protocol, which requires the *BRAS* to perform *Layer*

[1] When present, residential gateways are always routers.

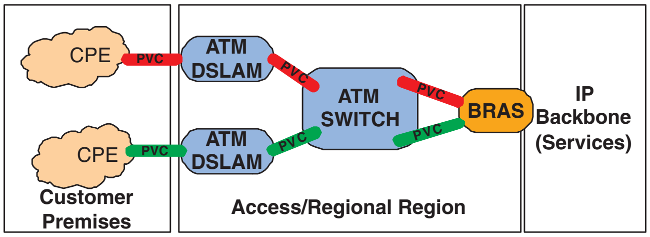


Fig. 2. DSL Forum model TR-59, ATM as aggregation technology.

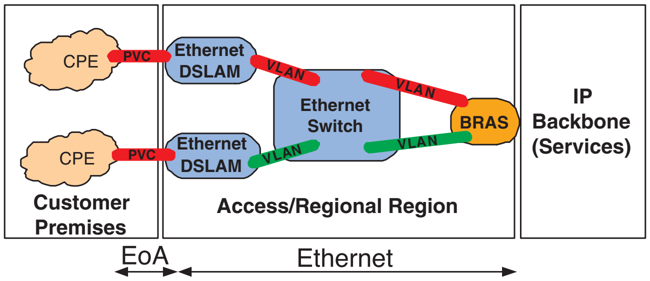


Fig. 3. TR-59, Ethernet over ATM.

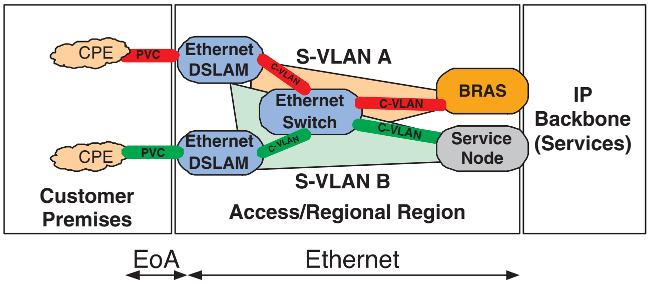


Fig. 4. TR-59, pure Ethernet concept.

2 Tunneling Protocol (L2TP) concentrator functions. On the other hand, if the aggregation is performed at the IP level, then the BRAS becomes a *PPP* terminator: PPP sessions are terminated and *IP* assignment is performed to re-route the traffic to the correspondent SP(s).

BRAS-centric architectures hold several drawbacks when it comes to *IP*-based services. A first drawback is that all the IP traffic has to go through the *BRAS*, independently of the physical location of the involved devices/entities, namely, end-users and/or *SPs*. For instance, *Peer-to-Peer (P2P)* traffic involves both sources and destinations which are within the CP region and yet, such traffic has to cross the entire access region. Given that the BRAS has to cope with a high number of complex functions, BRAS equipment is usually expensive, impacting on the scalability of the deployed architecture. A second drawback is the lack of proper multicast support: ATM is a connection-oriented, point-to-point (1:1) technology, while multicast requires a connection paradigm capable of supporting (at least) point-to-multipoint (1:N) transmission models. To give a concrete example of the possible problems that may arise, services such as *Internet Protocol TV (IPTV)* which require efficient multicast support on the access/aggregation region rely on the utilization of at least two different *Virtual Circuits (VCs)* allocated to multicast traffic per end-user: one VC per channel (multicast stream) and a special VC to support zapping (in practice, supported by means of the *Independent Group Multicast Protocol (IGMP)* [15]. Furthermore, there are some cases where bidirectionality is also required. Bidirectionality implies the replication of channels per end-user at the *BRAS*, resulting in additional overhead in the AN, and significant bandwidth overload across the aggregation region.

If Ethernet is used instead of *ATM*, then its connectionless nature and the ability to automatically support multipoint-to-multipoint connectivity (N:N) is the first step to allow BRAS decentralization and to explore better support for services such as multicast. While this potential is in fact being considered, a global deployment of a MAN core based on Ethernet as a single step is highly unlikely to be achieved due to cost reasons. Two main possibilities are therefore being considered for DSL infrastructures: to perform a global upgrade to Ethernet, or to deploy instead *Ethernet over ATM (EoA)* concepts. These are also the approaches followed by the *DSL* Forum which considers, as a first evolutionary step for the TR-59 model, the use of *EoA*. The resulting scheme is illustrated in Fig. 3, where the aggregation region incorporates Ethernet switches (instead of ATM switches). Then, the end-user PVCs are mapped on the DSL line directly to Virtual Local Area Networks (VLANs), being Ethernet frames transported on the PVCs between the CPE and the access region. Even though this approach does not take full advantage of Ethernet plug&play capabilities, it provides cheap bandwidth and operational savings: there is a one-to-one mapping to ATM's capabilities. The flip-side is that the whole network functionality is still centralized at the BRAS, thus being all the mentioned problems of ATM-based infrastructures inherited, despite the possible Ethernet advantages.

The second step considered by the DSL Forum for the evolution of the TR-59 model is the complete substitution of ATM by Ethernet [12], as illustrated in Fig. 4. What this step introduces is the capability to support configuration per service - *Service VLANs (S-VLANs)*- together with the support of individual (per end-user) policies. Furthermore, the BRAS functions can now be moved to other locations, as illustrated by the use of a specific *Service Node*. It should be noticed that the role of Service Node is simply a logical one. Such decentralization gives the support for better traffic differentiation and treatment. For instance, service selection and upstream policy enforcement functions which as of today are placed in the BRAS can be moved to the ingress of the access network, thus possibly allowing better control (e.g., prevention of malicious traffic). Placing service selection at the border of the access network allows it to be triggered earlier and to better aggregate traffic, improving resource provisioning and consequently, helping in reducing associated costs. Upstream policy enforcement at the ingress helps in avoiding or allows to better deal with bottlenecks, which drastically improve the behavior of applications with bidirectional requisites.

The next section summarizes the advantages and challenges that Ethernet faces in the MAN core, when compared to ATM based solutions.

A. Advantages Compared to ATM

Pushing Ethernet into the MAN core results in a more homogeneous transport infrastructure, which brings in little protocol overhead, low protocol conversion, and a better interface between access/regional networks. As a possible aggregation technology, the main advantages of Ethernet in comparison to *ATM* can be summed up as:

- **Better quality/cost trade-off**. While ATM is a powerful technology capable of providing support for the most varied services, ranging from regular voice to IP based services, ATM equipment is expensive and an optimal deployment of the transport core requires planning in advance. In contrast, Ethernet equipment is cheap and due to the large number of different rates and interfaces supported, the trade-off between cost and quality provided is better for Ethernet.
- **Higher flexibility**. ATM lacks flexibility when it comes to IP services. This is mostly due to its connection-oriented nature, which requires configuration to be provided statically. On the transport, whenever PPP is used to transport IP, IP information cannot be considered. Thus, the use of services such as IP multicast result in bandwidth losses and in lower aggregation efficiency.
- **Less overhead**. The connection-oriented nature of ATM and the limited frame size of 48 bytes makes it necessary to fragment IP datagrams, contributing to the traffic overhead. Total overhead on ATM backbones typically comes in between 15% and 25%. On a 155 Mbps circuit, effective throughput can drop to 116 Mbps [16]. In contrast, Ethernet brings in the IP adaptability already proven in LAN environments.
- **BRAS decentralization**. By decentralizing current BRAS functions, Ethernet provides the means to better aggregate (and differentiate) traffic, to optimize the transport of IP-based services, and to lower long-term expenses related to backbone equipment.
- **True multipoint-to-multipoint connectivity**. Given that ATM-PVCs represent point-to-point connections, in order to emulate point-to-multipoint or multipoint-to-multipoint connectivity between different sites it is necessary to perform provisioning of the multiple point-to-point PVCs and also, to establish IP routing on these *PVCs*. In contrast, Ethernet supports native multippoint-to-multipoint connectivity natively.

B. Challenges

When applied to the MAN core, Ethernet faces several challenges, being the main:

- **Reliability**. The Ethernet forwarding ability is based on spanning-tree approaches, which give a simple means to prevent information inconsistency by means of preventing topological loops: Ethernet avoids loops by blocking links. While this approach guarantees the delivery of data, in case of topology changes Ethernet may take several seconds to converge. Therefore, reliability in Ethernet is based not only on its intrinsic forwarding features, but also on traffic-engineering solutions which help in the control of provisioning of traffic by means of external (and manual) topology optimization, according to the specific needs of services and end-users.
- **Scalability**. Ethernet scalability problems arise from the fact that bridges learn *Media Access Control (MAC)* addresses promiscuously, i.e., they listen to every incoming packet learning MAC source addresses. While simple, the problem with this solution is that bridges learn every possible MAC address. Transposed to the Metro core this would result in core switches having to learn thousands of MAC addresses and having to deal with the corresponding MAC table load. This scalability issue is commonly referred to as *MAC address table explosion*. Adding to the learning overhead imposed by the basic promiscuous learning mechanism, Ethernet forwarding state is created on-demand, by performing *flooding*. In other words, whenever a switch needs to learn the direction (association to port) of a possible destination MAC address, it broadcasts the data packet which holds such MAC destination address on all of its ports (except the one where the packet was received in).
- **Resilience**. Resilience is one of the factors required to provide some guarantees to end-to-end services. Given that Ethernet is a *Best Effort (BE)* technology and despite the fact that an external QoS solution can be applied, Ethernet requires mechanisms capable of providing resilient networks, such as the ability to automatically detect node failures and to automatically perform network restoration. Bridging is usually an undermining factor to high availability especially in metro areas, due to the inherent topologies and to traffic load. Consequently, resilience in Ethernet is an aspect that is normally dealt with by means of traffic-engineering (e.g., *MPLS*, *Link Aggregation (LAG)*). For the specific case of ring topologies, there are solutions such as EAPS or ERP. A resilience analysis would require an extensive overview by itself and therefore this topic is left aside from the current paper, given that the goal is to focus on the forwarding mechanisms that Ethernet can rely upon. Further details concerning Ethernet and resilience can be found in related work such as [17].
- **Service differentiation**. Ethernet faces several problems concerning service differentiation per subscriber, given that there is no in-band signaling defined for resource reservation and therefore, some form of static controller is required to provide resource reservation and admission control. Usually, VLANs can be engineered to provide maximum bandwidth by means of VLAN Identifiers (VIDs), the IEEE 802.1p priority pair, and the *Differentiated Services Codepoint (DSCP)*, thus creating an overlay of provisioned pipes. Still, while resources are ensured, they are not optimized: some services mapped onto the same VLAN may still require specific guarantees, e.g., low delay/jitter, expected throughput. To cope with service differentiation, the operator has to be able to properly provision resources with fine-granularity, e.g., per session. Admission control and policy enforcement, as well as dynamic provisioning can be taken care of through the use of a static resource controller that can interact with the network elements. These limitations

have to be considered and overcome when devising Ethernet based services.

C. Services

In what concerns Ethernet services, conceptual guidelines are mostly being devised on the core of standardization entities such as the *Institute of Electrical and Electronical Engineers (IEEE)* [18], the *Metro Ethernet Forum (MEF)* [19] and the *Internet Engineering Task Force (IETF)* [20]. While IEEE standards are related to *Operation, Administration, Maintenance (OAM)* and in providing backward compatibility to current Ethernet standards, both the MEF and the IETF aim at providing intra-provider service definitions and inter-working support for Ethernet services. These approaches can be combined to create the most varied *Virtual MAN (VMAN)* services, as explained in the next sections, where an overview of the most interesting concepts is provided.

1) MEF *Service Definition - E-LINE, E-LAN, E-TREE:* In an attempt to take advantage the most from Ethernet flexibility, the *MEF* has been defining different categories of Ethernet services:

- *Ethernet Line (E-LINE)*. This is the regular point-to-point service, unidirectional and/or bidirectional. *E-LINE* can be used to provide services such as a connection between two sites in different cities, similar to a private leased-line service.
- *Ethernet Tree (E-TREE)*. As the name points own, this is the category of point-to-multipoint services. An *E-TREE* is an unidirectional service similar to *Ethernet Passive Optical Network (EPON)* as described in [21]. Both root-to-leaf and leaf-to-root directions are considered.
- *Ethernet LAN (E-LAN)*. *E-LAN* is a more powerful concept of an Ethernet service given that it allows creating multipoint-to-multipoint connection between different sites, where the addition or the removal of one site does not require re-configuring to the established *Ethernet Virtual Circuit (EVC)*[2].

2) IETF Service Definition - EoMPLS*,* VPWS*,* VPLS*, and H-VPLS:* While the MEF is defining the categories of services that ME can support overall, the IETF deals with the specific transport (and application) of Ethernet services in *Packet Switched Networks (PSNs)*. The IETF relies on the concept of a connection between two *Provider Edges (PEs)* nodes, the so-called *Pseudowire (PW),* which is used to transport *Packet Data Units (PDUs)* across IP/MPLS networks. The setup of the PW can be performed manually, by means of the *Border Gateway Protocol (BGP)*, or by means of the MPLS *Label Distribution Protocol (LDP)* [22]. Multiple PWs are transported inside a PSN tunnel, which can be generated using *Global Routing Encapsulation (GRE)*, L2TP, or MPLS. The PSN tunnel is used to "hide" Layer 2 information. For instance, if the core is IP/MPLS, only the PEs routers are aware of the creation of PWs and of the mapping of Layer 2 services to specific PWs; the remainder routers simply provide IP forwarding, or MPLS functionality between edges.

The transport of Ethernet frames can be based on L2TP (for IP), *Ethernet over MPLS (EoMPLS)* [23], or Layer 2 Virtual Private Networks (VPNs). While the former two solutions address the creation of a point-to-point connection service known as *Virtual Private Wire Service (VPWS)*, the latter embodies a concept known as *Virtual Pprivate LAN Service (VPLS)* [24]. VPLS provides the means to connect several sites (VLANs) into a single VLAN (a single bridged domain) over a provider's core. The VLANs specification defines the PE element as an edge-node capable of learning, bridging and replicating on a per VPLS basis. PEs that participate on the same VPLS are connected through a full mesh of *Label Switching Path (LSP)* tunnels. Multiple VPLS can be offered over the same set of LSPs. Signaling as specified in [24] is used to negotiate a set of ingress and egress VC labels on a per service basis. These labels are used by the PE to de-multiplex traffic arriving from different VPLS through the same set of *LSPs*.

Another IETF approach being considered for the transport of Ethernet services is the *Hierarchical VPLS (H-VPLS)*, which builds on LDP-based VPLS and enhances it with several operational and scaling advantages. H-VPLS can be applied in cases where it is desirable to extend the VPLS tunnels beyond the PE devices, e.g., into the premises of a *Multi-Tenant Unit (MTU)*: the MTU devices is treated as a regular PE and LSP tunnels are established also taking into consideration this new element. Thus, the VPLS core PW (IETF term: *hub*) are increased with the access PWs (IETF term*: spokes*). This creates a two-tier architecture, thus eliminating the need for a full mesh of PWs and consequently, reduces the signaling required. H-VPLS also enables VPLS-based services to span across multiple metro networks: a spoke is used to connect two different VPLS (in two different metro networks); in its simplest form, the spoke is simply an LSP tunnel. A set of ingress/egress VC labels are exchanged through this tunnel. The PEs treat the tunnel as they would treat a regular access PW. Thus, H-VPLS reduces the required inter-provider signaling and avoids the need for a full mesh of VCs and LSPs between the e.g., two MANs.

D. Achieving Scalability: Traffic Segregation and Control

While the mentioned services being defined attempt at taking advantage of the flexibility that Ethernet introduces, the underlying plug&play facet of Ethernet does incur scalability problems when applied to the MAN. This is due to the fact that Ethernet relies on 1) flat addressing and 2) address resolution based upon broadcasts. The addressing scheme in Ethernet is flat in the sense that each device has a unique and immutable identifier (address) which has no relation whatsoever with the geographic location of the device: MAC addresses are built upon the concatenation of 24 bits which identify a specific vendor - K and 24 bits which are assigned randomly to the interface by its vendor -*Network Interface Card (NIC)*. Ethernet bridges learn (source) MAC addresses automatically when receiving frames, associating the learnt MACs with a possible *direction* (port). Without adequate control, the

[2] The MEF defines an EVC as an association between two or more User-to-Network Interfaces (UNI). This is a tunnel that not only provides support for the transmission of Ethernet frames, but it also provides data privacy and security levels similar to the ones of ATM PVCs.

learning may originates MAC address table explosion (cf. section II-B).

The other mentioned aspect is the broadcast-based address resolution on Ethernet. When a frame with an unknown (not yet learnt) destination MAC address arrives to a bridge, then the bridge sends the frame on all its forwarding ports except the port where the frame was received at, i.e., the bridge broadcasts the frame. This allows, on the one hand, for a bridge that is aware of the destination MAC address whereabouts to react quickly (thus the data plane is minimally affected), but on the other hand broadcasts significantly consume bandwidth and result in sub-optimal network resource utilization. Consequently, Ethernet requires the application of some form of *flooding control* and of *traffic segregation* techniques to scale in MAN environments.

Traffic segregation is normally performed by means of VID tagging schemes [25]. This allows to split traffic into smaller, completely independent broadcast domains, but requires proper configuration in every participant networking device and does nothing to reduce the required MAC address table size. Furthermore, the use of VLANs is limited by the size of the VID tag, currently of 12 bits. A maximum of 4094[3] tags is possibly not enough, particularly for cases where traffic segregation is performed per end-user (one VLAN per end-user). This topic is further addressed next, in section II-D1.

Another way to perform traffic segregation is to split the aggregation area into several Ethernet *islands*. The advantage of relying on aggregation splitting is that it automatically reduces the *MAC* table size. The size of an island can be determined by the scalability of the used Ethernet switches, the number of concurrent sessions and the number of aggregation networks per IP edge. However, the drawback of this approach is complexity, given that it increases the required number of interoperability points and given that it requires careful manual intervention.

1) Stacking Schemes: Stacking (also known as *encapsulation*) schemes help to cope with the current limitation on the VID tag size: they provide the means to extend the 4094 stacking limit, through the encapsulation of tags. The *Q-in-Q (QiQ)* [25] technique provides VLAN-in-VLAN encapsulation, i.e., within a single provider's domain, there can only be 4094 simultaneous VLANs, but each of these VLANs can be further split into 4094 sub-VLANs.

VMAN *tagging* identifies uniquely a VLAN through the combination of the two VID fields, resulting in a maximum of VLAN different identifiers which the provider can control.

While QiQ is backward compatible with standard bridges, a VMAN-based solution is not. Additionally, both the QiQ and the VMAN approaches aim at providing scalability in terms of VLANs, but do little to limit the size of MAC address tables that bridges have to deal with. This is exactly what *MAC-in-MAC (MiM)* [26] targets. This encapsulation scheme hides, through the provider's core, customer VLAN frames by mapping them to PE nodes. This implies that PE nodes require more intelligence - they must keep state concerning the mapping of the customer VLANs and have to insert the provider MAC source and destination address in frames - but reduces the size of the MAC address tables in core switches, given that they only need to learn the source and destination MAC address of PEs. A specific application of MiM is described in [27].

These are the basic techniques used for stacking but as it will be discussed ahead in this paper, today the Q-tag placeholder is used in a way that allows some approaches to take advantage of its fields without jeopardizing communication with the regular type of Ethernet devices.

2) Controlling Multicast Traffic: IP multicast is a key feature for video distribution, given that it provides the ability to efficiently distribute information to a large number of subscribers. Multicast traffic is treated in Ethernet as broadcast and as such, multicast forwarding is performed by flooding. In other words, frames with a multicast MAC address as destination are sent to all ports of a switch (except the one on which the frame was received), as a regular broadcast packet. The main difference to a frame destined to the broadcast address is that only the switches that have registered to that multicast group will in fact acknowledge such frame content - the others simply discard it. This has several consequences which mostly impact on the scalability factor and the bandwidth usage efficiency of the access/aggregation region.

In what concerns the transport of IP multicast across Ethernet regions, it is not enough to perform a direct mapping between the IP multicast addresses and the Ethernet addresses, given that IP and Ethernet addresses hold different sizes, namely, 32 bits for IP version 4 (IPv4) and 48 for Ethernet: from the 28 less significant bits of an IPv4 multicast address, the 23 lower bits are directly mapped to the lower bits of the Ethernet EUI-48 [28] MAC address. The remainder 25 higher order bits of the group MAC address are statically assigned to the prefix 01:00:5E. Therefore, there are 5 bits from the IPv4 address that cannot be mapped, which leads to 32:1 possible collisions.

The situation is even worse if IP version 6 (IPv6) is considered. Instead of relying on the EUI-48 MAC address format, IPv6 relies on the EUI-64 MAC address format (a basic requirement for the support of autoconfiguration) and therefore, now the 32 less significant bits of the IPv6 address overwrite the 32 less significant bits of the EUI-64 address. This simplifies the mapping, but does not avoid the collision problem that already occurred in IPv4. Furthermore, IP to Ethernet multicast mapping collisions are also a result of the option taken in terms of the IP multicast routing protocol chosen for distribution, choice which normally goes to the *Protocol Independent Multicast-Sparse Mode (PIM-SM)* [29]. If such choice goes instead to the *Protocol Independent Multicast Source Specific Multicast (PIM-SSM)* [30], then there is an additional piece of information that is lost, i.e., the mapping to the IP multicast source. Therefore, IP multicast cannot be supported by direct mapping to Ethernet multicast. Instead, there is the need to couple multicast support with flood control techniques that range from simple filtering to the more complex deployment of specific protocols. The non-proprietary and basic techniques that can be considered when deploying multicast services on Ethernet are:

- *IGMP/Multicast Listener Discovery (MLD)* [31] **(trans-**

[3] With 12 bits, the number of possible VLAN-IDs is $2^{12} = 4096$tags. However, two IDs, 0 and 4096, are reserved.

parent) snooping [32]. On a specific multicast VLAN, all the involved switches filter IGMP (for IPv6, MLD) packets to obtain group membership multicast and to prevent flooding. The advantages of IGMP/MLD snooping are first its simplicity, and second, its ability to direct multicast streams to the adequate subscriber ports. The drawbacks ofthis solution come from the fact that high volumes of data give rise to a heavy computation price, given that every switch on the path must snoop IGMP/MLD packets. IGMP/MLD snooping is completely transparent, in the sense that it does not require modifications to the IGMP/MLD messages.

- **IGMP proxying** [33]. Usually applied in routers that do not support multicast, IGMP/MLD proxying is another technique also commonly used in the access/aggregation region. For instance, the AN becomes an IGMP "relay", being able to determine and map multicast membership, and communicating that information directly to the proper EN (e.g., BRAS). Given that this technique aggregates IGMP requests - IGMP joins and leaves are translated into a single request each -, it reduces the required signaling on the access/aggregation region. However, it is not transparent in the sense that it usually required modifications to the IGMP message, e.g., client IP address.
- **Multicast VLAN Registration (MVR)**. MVR is a technique specifically designed to allow the widescale deployment of multicast traffic (e.g., broadcast of TV channels) on ring topologies. MVR provides the means to create single multicast VLANs that can be utilized by subscribers that are assigned to different VLANs. This means that multicast streams are sent in the multicast VLAN and still they do not affect the subscriber traffic belonging to other VLANs. Therefore, MVR prevents the duplication of multicast channels per subscriber. Even though independent from IGMP, MVR requires the switch to have IGMP snooping activated. It is therefore a technique that enhances IGMP snooping, and is specifically suited for support of massive video distribution services.
- ***Generic Attribute Registration Protocol (GARP)/GARP Multicast Registration Protocol (GMRP)/GARP VLAN Registration Protocol (GVRP)***[4] [34]. GMRP is an OSI Layer 2 protocol that has functionality similar to the one of IGMP/MLD snooping. It allows switches and end-hosts to dynamically register group membership information, according to services provided by GARP (which deals with provisioning attributes), and a way to disseminate such information across a specific VLAN. GARP provides specific VLAN support in the form of GVRP. A GMRP-based solution must consider support both on the switches and on the CPE, where it is used in common with IGMP. The access node receives both GMRP and IGMP information coming from the CPE. It then uses GMRP information to control multicast distribution at Layer 2. Specific VLAN configuration is provided by means of GVRP, which is a part of GARP. The major advantages of GMRP is that it reduces the overall effort associated with IGMP on the access/aggregation but it still requires IGMP support both on the CPE and access node. Due to the fact that it does not provide any advantage when compared to IGMP snooping, GARP/GMRP/GVRP deployment is not widespread.
- **Multiple Registration Protocol (MRP)**. To address some of the scalability issues of GARP, MRP [35][35] has been proposed to allow participants of a so-called MRP application to register attributes within bridged LANs. The standard currently defines two MRP applications, namely, *Multiple VLAN Registration Protocol (MVRP)* and *Multicast Multiple Registration Protocol (MMRP)*. MVRP is used for VLAN registration, while MMRP is used for group MAC address registration. As mentioned, the described flooding techniques are normally applied together with traffic segregation techniques, e.g. VLANs, to control multicast distribution. VLANs may be deployed to support the traffic related to a single subscriber *(Customer VLAN (C-VLAN))*, traffic related to a single AN and consequently to a specific group of subscribers (VLAN per AN) or be deployed to support traffic related to a single service provided, e.g., IPTV (*S-VLAN*).

III. IEEE Spanning-Tree Approaches

In this section we provide an overview of the current IEEE spanning-tree standards, namely, STP, RSTP, and MSTP, highlighting the major differences between these protocols. It should be noticed that currently RSTP superseeds STP. Consequently, STP is here presented simply in order to help to understand how these protocols evolved and what were the problems that lead to the appearance of each approach.

A. STP

Standardized in 1998 as IEEE 802.1D, STP relies on a minimum shortest-path spanning-tree to create a logical, loop-free tree structure that incorporates both segments and bridges. Being a minimum shortest-path spanning-tree, this tree is composed of shortest-paths from every node to the root, without any guarantee that a path between two nodes is a shortest-path.

STP appeared as a solution that would allow two different end-systems connected to two different LAN segments to communicate. The basic idea for such element was that it should passively listen to every packet sent - *promiscuous listening* - and somehow learn the location of the end-system. This is achieved by learning the association between the packet source MAC address and the port on which the packet is received. This association allows the forwarding of the packets in a very simple way without a need for some form of a hop-count. However, because bridges listen to every single packet they get, when loops occur (e.g., due to a topology change) there may be information inconsistency or duplication. Relying on a spanning-tree approach is therefore a simple way to prevent these problems (by preventing topological loops).

[4] Generic Attribute Registration Protocol (GARP)/GARP Multicast Registration Protocol/GARP VLAN Registration Protocol

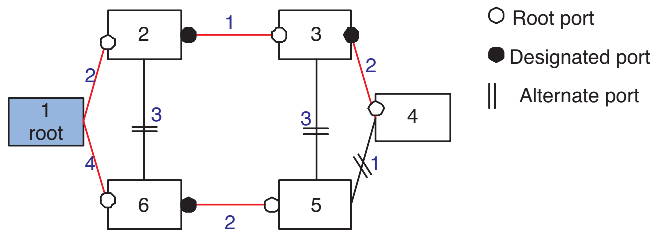


Fig. 5. STP example.

In terms of operation, STP goes through the following steps:

1) **Election of a root bridge**. Normally this is performed based upon static parameters, namely, the MAC address concatenated to the (variable) priority field - *Bridge Identifier (BID)*. The bridge that is represented by the lowest BID becomes the root of the logical spanning-tree. The root bridge location is crucial to a good behavior of the spanning-tree approach and as of today, the choice on which bridge to use as root is tuned manually.
2) **Computation of the minimum cost path from each non-root node to the root**.
3) **Designated port election**. For each network segment choose a port (*designated port*), on which the bridge is responsible for forwarding data. In other words, the bridge becomes a *Designated Bridge (DB)* for the segment attached to the port in question.
4) **Root port selection**. Choose a port (*root port*) that gives the best path from a specific bridge to the root bridge.
5) **Select the ports to include in the spanning-tree**. These are the root port plus any ports on which the bridge has been elected as DB.

An example of STP is provided in Fig. 5, where each bridge is represented by a square. When the bridges wake up, they exchange *Bridge Protocol Data Units (BPDUs)*. Each BPDU contains a BID which is used to select the root bridge. Then, for each non-root node STP allows to elect a root port and a designated port. The process of selecting designated ports is again based upon information contained in the BPDUs, which allows to compute the shortest-path from a bridge to the root bridge. The port associated with the link and that has the lowest link path cost to the root becomes the root port. STP breaks loops by deactivating some links , i.e., by blocking the ports associated to a link that are not root nor designated ports.

STP relies on two different types of BPDUs, namely, *Topology Change Notification (TCN)* and *Configuration* BPDUs. TCN BPDUs are exchanged by bridges when a topology change is detected. The root bridge then has to notify bridges of the change. This is performed by having the root bridge setting up a *Topology Change (TC)* flag in every BPDU it sends, for a period of *Forward Delay* + *MaxAge* (15+20=35 seconds by default).

Configuration BPDUs are only exchanged by the root bridge every *Hello* time (default of two seconds) and carry the required information to recompute the spanning-tree. Regular bridges receive Configuration BPDUs on their root ports and forward them on the designated ports.

Once the logical **or active** topology is established, STP monitors the topology for possible topology changes. Events that may trigger topology changes are link/node failures, addition/removal of new links/nodes, or change of bridge configuration.

After a topology change, STP steps have to be re-computed. We name this procedure *reconvergence*. STP re-convergence may take minutes depending on the assumed topology, being these values unacceptable within the MAN context.

As an answer to the re-convergence times of STP, RSTP has been proposed by the IEEE.

B. RSTP

RSTP was introduced in IEEE802.1w as an amendment to IEEE 802.1D and due to its popularity is now a part of 802.1D. RSTPbuilds upon STPand provides faster re-convergence, theoretically lower than one second. RSTP and STP are quite similar in operation, being RSTP in practice simply an optimization of STP. Main characteristics of RSTP are:

- **BPDU simplification**. Instead of using two different types of BPDUs, RSTP only relies on a single type, which is similar to the STP *Configuration BPDU*, where the version number is set to two. In addition to the two types of flags STP uses in topology changes, namely, *Topology Change* and *Topology Acknowledgment,* RSTP uses six additional bits to encode the role and the state of the port originating the BPDU, as well as two flags to handle the proposal/agreement mechanism.
- **Faster filtering database aging**. In STP, the MAC-to-port entries that compose a *Forwarding Database (FD)* are not flushed. Instead, TCNs are sent to the root bridge which then again sends BPDUs to notify other nodes about the change detected. In RSTP, the switch that detects a topology change automatically sends a BPDU with the TC flag on to other switches, and automatically flushes its FD.
- **Simplified negotiation process between bridges.** In STP, bridges do not generate their own BPDUs - they simply relay BPDUs from the root bridge. Consequently, to know that the root bridge is down, a bridge has to rely on not having received a BPDU for *MaxAge Time* (by default, 20s) to then trigger the process of a new root election. In contrast, RSTP switches expect to receive a BPDU (from another switch) within three Hello times. If no BPDU is received, the switch assumes that connectivity to the neighbor is lost.
- **Simplified STP state machine**. The number of port states to three (instead of the five from STP, cf. Table I);
- **Differentiation between regular and edge ports**. RSTP allows to configure ports that connect to end-hosts as *edge ports*. These ports do not need to transition through the regular three states: they are automatically set to forwarding state. If a BPDU is detected on an edge-port it automatically becomes a non-edge port.
- **Handshake mechanism to speed up link failure re-convergence**. This is in fact the main difference from RSTP to STP and the enabler of the faster convergence, as explained in the next section.

TABLE I
STP VS. RSTP PORT ROLES.

STP port role	RSTP port role	Port active?	Port learning MACs?
Disabled	Discarding	No	No
Blocking	Discarding	No	No
Listening	Discarding	No	No
Learning	Learning	No	Yes
Forwarding	Forwarding	Yes	Yes

1) Handshake Mechanism for Faster Link Failure Re-convergence: To provide a basic comparison between the operation of RSTP against the one of STP, Fig. 6 illustrates a topology with four bridges, being bridge 1 the root. As illustrated, it is assumed that the link connecting bridge 1 to bridge 4 fails. With STP, the time to achieve re-convergence would be around 50s, due to the following:

- bridges 3 and 4 would wait *MaxAge* seconds (by default, 20 seconds) before aging out the respective entries. During that time they continue to forward information on the old path.
- After this interval, bridge 3 realizes (by means of the alternate port state) that there is another possible path to the root, i.e., port 02. It selects this port as root port and advertises it to bridge 4 by means of port 01, which becomes a ddesignated port. Bridge 4 detects the topology change and changes port 02 to Root port.
- During the topology reconfiguration and to prevent information inconsistency, bridge 4 puts ports 01 and 02 first in learning state (15s), and then in listening state (15s), resulting in an additional 30s delay.

Assuming that the link gets restored, then bridge 4 starts receiving again BPDUs from bridge 1. Consequently, bridge 4 elects again port 01 as a root port and 02 a designated port. Port 01 has to transition through listening and learning state before data is forwarded by it. Moreover, port 02 is again changed to designated. The same time of operation happens in bridge 3.

This process is faster for RSTP. When the link between 4 and 1 fails, then bridge 4 automatically announces itself as root. To simplify the example, we assume that bridge 4 has the lowest BID after bridge 1. Bridge 3 receives such information and recognizes that the connection to the root bridge (it has in stored, bridge 01) is down. Consequently, it elects bridge 4 as its root bridge, transitions port 02 to root port and immediately places it in forwarding state. The data sent allows bridge 4 to transition port 02 to root. Then, switch 3 performs a *sync operation* with 4 to transition port 01 to forwarding state. This sync operation relies on exchange of BPDUs, but requires no additional timers. In addition, bridge 02 is still connected to bridge 01. Consequently, bridge 02 sends a proposal to bridge 03 stating that 01 is the root. Bridge 3 again blocks its port 01, sends an agreement to bridge 02 and a proposal to bridge 4. Bridge 4 sends an agreement to bridge 03. Bridge 03 puts its port 01 into *designated-forwarding*: the logical topology is set again. Consequently, agreeing on a new topology requires less than 1s with RSTP.

Assuming that the link gets restored, then when bridge 1 detects the link is up it starts a sync process with bridge 4 to transition port 01 to forwarding state, i.e., bridge 1 sends a BPDU with a proposal flag set. Bridge 4 realizes that this path is the shortest-path to the root and asserts the sync, i.e., it makes all non-edge designated ports transition into blocking mode. Then bridge 4 acknowledges the proposal and consequently, bridge 1 transitions port 1 to forwarding. Having this being solved, there is no the need to break the loop between bridges 4 and 3, which repeat a similar process. Then, the same process has to be repeated between bridges 3 and 2.

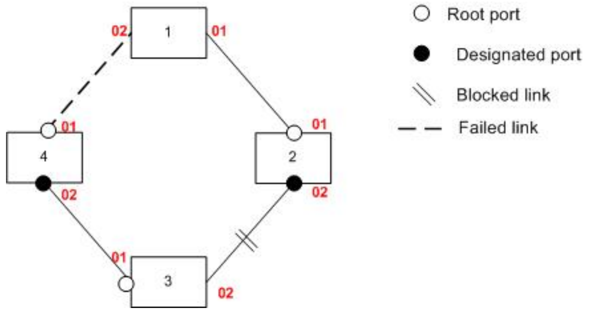


Fig. 6. Example of link failure.

This implies that while RSTP takes the same setup time as STP, a link failure restoration is quite fast. Nonetheless, RSTP re-convergence performance is affected by:

- the complexity of the network;
- the limit of BPDUs that can be exchanged for network stability;
- the failure location in comparison to the root location.

While RSTP improves the spanning-tree re-convergence times, depending on the parameters mentioned it can still take several seconds to converge in specific cases as explained in [36] [37]. Two major problems contribute to this:

- **Count-to-infinity** [38]. When a root bridge failure happens, RSTP may take several seconds to converge (5s). The count-to-infinity behavior (cf. [38]) can occur when the root fails and the resulting reconfiguration holds a loop. If BPDUs destined to the old bridge are on the network, they may be continuously flooded. The loop will end when the old root's BPDU *MessageAge* reaches *MaxAge*, which only happens after *MaxAge* hops.
- **Port role negotiation**. To prevent loops, RSTP negotiates every port transition. port negotiation is performed hop-by-hop in case of link failure, as illustrated in Fig. 7, for a ring topology (worst-case scenario). In the illustrated scenario, the link closer to the root bridge fails, triggering the topology reconfiguration. Consequently, all the traffic needs to be redirected: consecutive bridges on one side of the ring exchange port roles. The port role exchange is explicitly signalized by both bridges, to prevent loops. But, if both requests arrive simultaneously, the bridges may end-up in deadlock negotiations, in which case the reconfiguration will take 6s (the time required for the bridges to re-send requests). Another limitative factor is the rate limit which is applied in case of re-configuration. This limits the sending of BPDUs to one per second, per port, which may delay the convergence. However, the transmission can be set up to 10 BPDUs per second. In addition, it should be noticed that port role negotiation

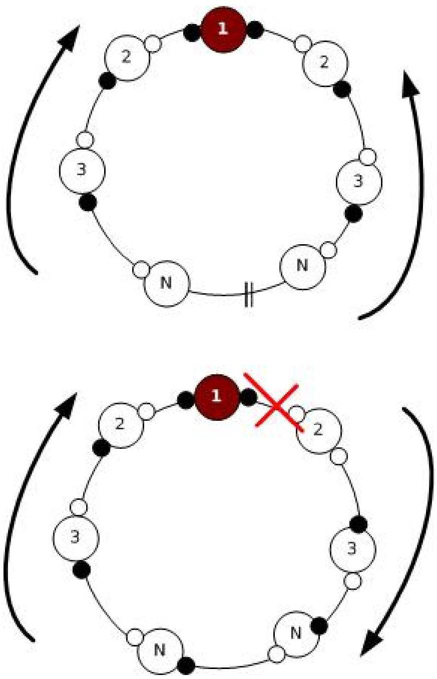


Fig. 7. Port role negotiation example.

may result in large reconfiguration delays if and only if the root bridge is involved in the failure.

C. MSTP

MSTP [39], originally defined in IEEE 802.1s as an amendment to IEEE 802.1q and now integrating this standard, aims at providing a solution for the scenario that STP cannot contemplate, i.e., having VLANs that cover the same network elements being each assigned to a different spanning-tree. In STP, each VLAN corresponds in fact to a spanning-tree. Consequently, blocked links for a VLAN cannot be used for another, as illustrated in the case of Fig. 8, where it is only possible to establish a single VLAN (VLAN1) between bridges 1 and 2. This means that despite the fact that two links are available between 1 and 2, only the link $(1/1, 2/1)$ can be used.

With MSTP, both links can be active at the same time. MSTP works by providing instances of a same spanning-tree, onto which VLANs can be mapped. MSTP provides therefore the notion of *Multiple Spanning Tree Region (MST),* a region that comprises several VLANs. Inside a MST there is a single *Internal Spanning Tree (IST)* and several (more than two and no more than 64, according to IEEE 802.1s) *Multiple Spanning Tree Instance (MSTI)*. In practice, the IST corresponds to the regular spanning-tree (in the MSTP case, obtained by running RSTP), and by default all VLANs in the region are assigned to the IST. MSTP provides the means to assign some of such VLANs to MSTIs, therefore obtaining better bandwidth efficiency - links blocked in an instance may be active in other instances. The IST is used to channelize information concerning the remainder instances.

MSTP uses specific MSTP BPDUs to perform global control by means of the IST. Inside a specific MSTI, *M-records* (record containing information specific to a MSTI, e.g., root) are appended to BPDUs. When a BPDU leaves a MST region (by means of the IST), the M-records are removed, being the regular RSTP BPDU sent on the IST. So, inside a MSTI, bridges run RSTP automatically.

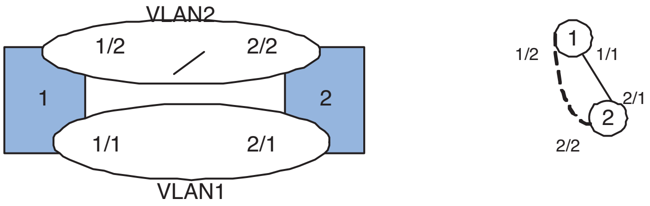


Fig. 8. VLAN blocking due to mutual spanning-tree.

The different MSTs are interconnected by the *Common Spanning Tree (CST)*. Additionally, the Common Internal Spanning Tree (CIST) connects all the ISTs and the CST together. In practice, each MST corresponds to a logical region (administrative region), and each switch belonging to a specific region holds the following attributes:

- an alphanumeric configuration name (32 bytes);
- a configuration revision number (two bytes);
- a 4096 element table that associates each of the 4096 VLANs to a given instance.

The obvious advantage of MSTP is that it allows to have multiple paths to the same destination(s). This means not only better bandwidth efficiency but also the opportunity to implement load-balancing. However, MSTP is not trivial to configure and in fact manual configuration (or some sophisticated external tool) has to be used to properly configure all the elements.

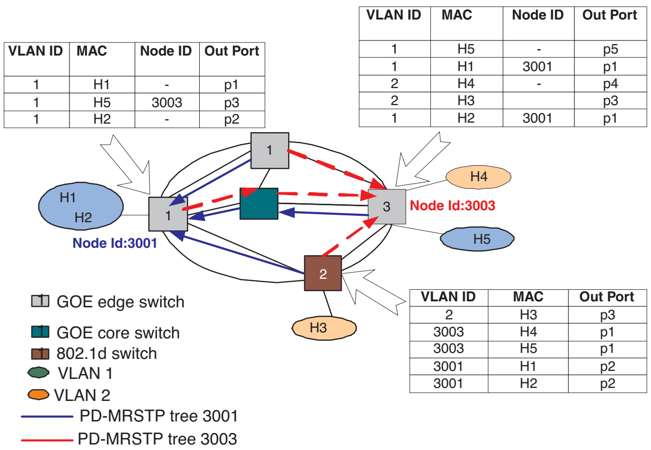
In addition, several works look into possible optimizations of MSTP. For instance, [40] looks into possible optimizations taking QoS into consideration. [41], [42] proposes an algorithmic approach for constructing multiple spanning tree regions having as focus enterprise network domains and as evaluation parameters convergence times and scalability in terms of VLAN IDs, as well as broadcast domain size reduction.

IV. Novel spanning-tree Based Approaches

While the de-facto Ethernet forwarding protocol is RSTP, within the MAN it is clear that there are still some issues mostly related to resilience and to convergence that significantly affect the performance of Ethernet. Consequently, several works attempt at providing enhancements still building upon spanning-tree approaches, as explained in this section.

A. GOE, Global Open Ethernet

Global Open Ethernet (GOE) [43] is an advanced Ethernet approach that relies on a proprietary spanning-tree solution named *Per-Destination Multiple Rapid Spanning Tree (PD-MRSTP)*. GOE splits the functionality of bridges between bridges at the edges - *edge bridges* - and at the core - *core bridges*. By means of PD-MRSTP, GOE automatically creates a tree instance for each edge bridge. Not only are these spanning-trees, but for unicast traffic, they also represent *sink-trees*, as illustrated in Fig. 9, where red (dashed) arrows represent the sink-tree with root bridge 3. Consequently, when



VLAN ID	MAC	Node ID	Out Port
1	H1	-	p1
1	H5	3003	p3
1	H2	-	p2

VLAN ID	MAC	Node ID	Out Port
1	H5	-	p5
1	H1	3001	p1
2	H4	-	p4
2	H3		p3
1	H2	3001	p1

VLAN ID	MAC	Out Port
2	H3	p3
3003	H4	p1
3003	H5	p1
3001	H1	p2
3001	H2	p2

Fig. 9. GOE operational example.

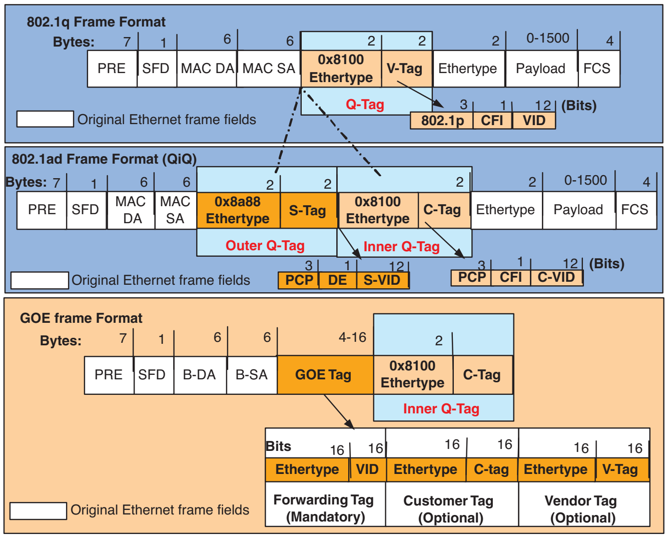


Fig. 10. GOE header format compared to 802.1q and 802.1ad frame format.

booting, every edge bridge creates a shortest-path to every other edge bridge.

To forward frames between the GOE bridges at the same time keeping backward compatibility, GOE relies on QiQ encapsulation where a special GOE tag is placed on the place of the outer Q-tag. The GOE tag format (cf. Fig. 10) is, in its mandatory form, equal to the regular QiQ and compatible with legacy bridges that implement 802.1q. In its optimized form (only understood by GOE bridges), the new tag may hold in addition a customer and a vendor tag. Both the tags incorporate the Q-tag format, i.e., 16 bits for the Ethertype and 16 bits for the tag information.

GOE also optimizes the forwarding plane. The forwarding tag, which to regular 802.1q enabled bridges looks like a regular Q-tag, contains as usual a VID. However, that VID identifies an egress edge bridge and consequently the adequate tree instance of which that bridge is the root. In other words, GOE uses VIDs to identify bridges (and not just ports). The GOE forwarding tables map MACs to the root node of each tree. Consequently, core bridges just have to rely upon VIDs to perform the forwarding (no need to look for a specific MAC address).When the frame reaches the root of the tree (egress edge bridge), the GOE tag is removed, and the packet is then forwarded to its destination, according to the MAC destination address. The GOE path learning mechanism is a distributed learning process, that relies on three different forwarding trees:

- GOE forwarding tree (sink, spanning-tree) for known traffic, which represents a sink-tree between GOE nodes and where the sink-tree is an edge GOE node;
- legacy spanning-tree, which is used to exchange traffic between the GOE nodes and legacy nodes;
- GOE source-tree (reverse tree of a GOE forwarding tree), used to broadcast unknown/multicast traffic.

Known traffic forwarding is performed on either the GOE forwarding tree, or on the legacy tree, depending on whether or not the first bridge on the path is a GOE node. Frames hold a GOE tag which is interpreted as a regular tag by legacy bridges. GOE bridges know whether the tag corresponds to a GOE tag or to a regular Q-tag, because the VID space is split into normal mode (1 to X) and GOE mode (X to 4095).

Each MAC host is associated with a VID, which is therefore inserted into the frames. Along the way, core switches just perform a tag lookup against the information kept on their forwarding tables.

The major difference between the GOE forwarding when compared to legacy forwarding is that while the latter is based on the VID and destination MAC address, the GOE forwarding simply relies on the VID (VLANs are unidirectional). This also means that the path between A and B is not necessarily the path between B and A.

When an entry for a specific MAC is not found, the corresponding bridge (S) forwards the frame through the GOE broadcast tree, specifying itself (using its identifier, I) as the root of the tree. The destination bridge (D) learns the relationship between the source MAC address, the source bridge identifier and respective port, and redirects the frame to the destination. When D gets a frame back (from the destination host), it finds an entry for the source host and pushes the corresponding tag onto the frame, now forwarding it on the GOE forwarding tree represented by the I identifier. When S gets the frame, it strips the tag and re-directs the frame to the destination.

The main advantages of GOE are:

- *root recovery is avoided*. Given that the root of each spanning-tree is also the destination bridge, there is no need to reconfigure the whole tree, given there is no alternative physical access point, unless users are connected to two different bridges, e.g., multi-homed scenario. For the latter, the forwarding can be recovered using another destination bridge;
- *in-service reconfiguration*. When a new bridge is inserted, instead of re-constructing a new spanning-tree, the GOE simply creates a backup spanning-tree, using the backup identifier. While the backup spanning-tree is being created, the old tree is used; it is up to the root to trigger the initiation of the new tree. This means that possible service interruption is reduced;
- *enhanced failure recovery performance*. The convergence

times claimed by the authors are in the order of **2ms**. It is claimed that this is due to the fact that the GOE forwarding table is a direct memory-mapped table, which directly associates VIDs with the internal memory address to resolve output ports for specific GOE tags;
- *less forwarding state kept*. Each entry kept on a bridge corresponds to a forwarding identifier (root of tree). In contrast, the regular operation of VLAN tagging, only up to 4094 entries are allowed per bridge. However, for large scale VPNs, the authors propose a hierarchical address, based on the standard VLAN stacking schemes, and on the use of the GOE header;
- *failure recovery time* is not dependent on the number of spanning-trees.

However, there are also some disadvantages:

- *scalability*. Given that GOE relies on the VID scheme to identify the roots of each spanning-tree, this only allows a maximum of 4094 trees to be created. Depending on the size of the provider, this may be too little. The stacking solution proposed by the authors allows this number to scale, but might increase the complexity of the lookup. Consequently, GOE may not be suitable to be applied in environments incorporating a large number of switches;
- *unidirectional* VLANs. The use of unidirectional paths between bridges implies asymmetric forwarding, i.e., the path from source to destination bridge will most likely not be equal to the path between destination and source. This is not backward compatible with legacy equipment, which creates bidirectional VLANs and therefore, requires special support from edge and core switches.

B. AMSTP

The *Alternative Multiple Spanning Tree Protocol (AMSTP)*[44] builds upon RSTP and MSTP by having each bridge on the network automatically owning its tree instance. In other words, AMSTP creates one *source-tree* per bridge i, being i the root bridge of tree i. The consequence of the utilization of source-trees is the ability to build shortest-paths between every bridge i, similar to GOE (where sink-trees are used).

AMSTP starts by building a global (standard) RSTP tree covering all the bridges and then building the remainder instances, one tree instance per bridge. For the remainder tree instances, each bridge elects itself as root of a tree instance. For that purpose, each bridge appends a new record, the *AM-record* to the main tree BPDUs information about its tree instance (claiming itself as root) and automatically accepts every other bridge as element of its own tree instance. It also accepts the claims of every other bridge as root of the remainder tree instances.

Each tree instance is identified by the MAC address of its root bridge and the remainder tree is built relying on the regular MSTP procedure, i.e., port selection according to the minimum path cost to the root, and port identifier for tie-breaking.

To provide backward compatibility with legacy bridges, AMSTP bridges exchange between themselves encapsulated frames. Ingress AMSTP bridges encapsulate the RSTP (or STP) frame adding an additional Layer 2 header containing as source MAC address the address of the current bridge, and as destination MAC address the address of the egress AMSTP bridge. MAC addresses are learnt by means of the received frames - the outer header provides information about the MACs of other AMSTP bridges, while the inner header contains the MAC destination addresses. The learnt MAC information is kept on a table (per port) and contains the MAC address pairs learnt at the port. When an AMSTP bridge wants to forward a frame, it checks first the destination in this table - the egress AMSTP bridge MAC address. If available, the AMSTP bridge learns also the port association to this MAC address, thus being able to forward the packet. In case the destination MAC address has not been learnt yet, then the frame is sent to a reserved Layer 2 multicast address which represents all the AMSTP bridges.

The type of messages is also very similar to MSTP: both comprise a BPDU with several AM-records prepended. AM-records are used to negotiate the tree instances, while BPDUs are used to set the trees and to negotiate possible port/role transition. The encapsulated frames are forwarded by means of the source-tree for which the ingress bridge is the root

While AMSTP is very similar in operation to MSTP, its main advantage is the use of source-trees which automatically enables shortest-path forwarding between advanced bridges, thus avoiding the complex configuration normally required in MSTP.

However, while MSTP gives the choice to the operator to determine the number of trees to configure. (1 to 64), AMSTP automatically creates N trees, being N the number of AMSTP bridges in the network. It should also be noticed that each of these trees corresponds in practice to a unidirectional VLAN, while normally VLANs are bidirectional.

C. Shortest-Path Bridging

Due to the realization of the drawbacks of spanning-tree approaches, several vendors within the IEEE 802.1 working group showed strong interest in shortest-path bridging. This lead to the creation of an amendment[45] to IEEE 802.1q currently still in draft format[5]. The underlying idea is to use a tree instance per bridge to be able to always rely on shortest-paths, similarly to what is performed in GOE (cf. section IV-A) or to *Routing Bridges (Rbridges)* (cf. section V-C).

The initial discussion was triggered by the parallel work (Rbridges) developed in the IETF *Transparent Interconnection of Trillions of Links (TRILL)* working-group[46], whose main purpose is to design a solution for shortest-path frame routing in multi-hop IEEE 802.1-compliant Ethernet networks using an existing link-state routing protocol (cf. section. V-C for further details). Consequently, the first proposal in terms on how to create the tree instances was based upon some form of link-state information exchange (*Intermediate System-to-Intermediate System (IS-IS)*[47] as proposed in Rbridges), similar to routing. In the current PAR, a source-tree instance is created per bridge. Relying on a routing approach to compute the different tree instances results in faster convergence and

[5] PAR has been approved until December 2009.

better bandwidth efficiency given that all links can be used and provides a quick propagation for the learnt MAC addresses.

The flip-side of considering routing approaches to provide the tree computation is that such approaches do not necessarily require symmetry. In Ethernet and when multiple tree instances are present, it is desirable for the path from a node A to a node B in the tree instance owned by A to be the same path in the tree instance owned by B, or MAC learning won't work properly.

This amendment is the realization that Ethernet would benefit from a shortest-path forwarding in addition to a dynamic path computation plane (control plane). Nonetheless, there are still several items being worked upon before the amendment can take shape in reality. The current discussion on the topic is being done in strong cooperation with the IETF working group TRILL, where Rbridges (cf. section V-C) is being developed. Furthermore, the routing direction to follow (link-state or distance-vector) is still not completely decided.

D. Viking

The Viking [48] approach aims at providing faster recovery times for STP by using *backup path selection in advance*. Its main goal is to provide load-balancing, by taking advantage of possible unused links between two end-points. Viking builds upon an MSTP proprietary implementation, Cisco's *Per-VLAN Spanning Tree (PVST)* [49], relying on the principle of computing multiple spanning-trees in order to re-use the different links between two different points. In other words, Viking **computes multiple spanning-tree instances between sources and destinations**. The goal is to have at least two different *switching* paths between every two end-points of a network. The choice on which path to use is based upon regular VID tagging: the set of possible switching paths between two nodes are incorporated into different spanning-tree instances. These instances are pre-computed and therefore, the traffic is easily diverted to the available switching paths.

The tag selection is performed by end-hosts and not by the switches, meaning that Viking extends the VIDs until the end-hosts. To fight back the scalability problems of VLAN stacking, Viking relies on an algorithm that minimizes the overall number of required spanning-tree instances while maximizing the number of active links. This is performed using the Ethernet traffic prioritization mechanism 802.1p together with VID selection: traffic corresponding to backup paths are given a lower priority than traffic corresponding to primary paths. Viking also holds the following assumptions:

- there should be at least two different switching paths per node pair in two different spanning-trees which do not share intermediate nodes, or links. This improves fault tolerance, given that it's the minimum condition to have two different paths between sources and destinations.
- the path selection is expected to maximize the utilization of marginally loaded links and minimize the use of heavily loaded links. This gives the means to provide adequate load-balancing.
- The spanning-tree instances should minimally overlap with each other. This gives the means to maximize the number of active links, thus improving bandwidth efficiency.

Viking does not run directly on the switches. Instead, it is an approach based on long-term monitoring and reconfiguration of the network to achieve load-balancing. In fact, Viking holds two different components: a client, the *Viking Network Controller (VNC)* resides on end-hosts, and a centralized manager - *Viking Manager (VM)* - is located somewhere on the network, e.g., a centralized server. The VNC performs several tasks such as load measurement, VLAN selection and respective VID tagging. The VM is responsible for traffic-engineering and for fault tolerance. It is also in charge of informing the VNC about VIDs to use either upon query or pro-actively, after a topology change. Consequently, the VM holds a global view of network resources (based upon information fed by the several active VNCs). These two components work as follows:

- the VNCs provide the monitored traffic information to the VM periodically. Based on such information, the VM obtains on the long-run global knowledge about pair-wise load statistics.
- The VM acquires the topology by an external means, e.g., regular topology characterization tools, or information entered manually by the operator.
- The VM relies on the load characterization provided and on the global topology perspective it contains to select load-balanced primary and backup paths between every two pair of end-hosts.
- The computed paths are then aggregated into different spanning-trees instances - different VLANs, according to some common properties, e.g., shared links, shared nodes, shared segments (the algorithm presented seems not to be completely efficient; it starts with longest paths so that it avoids unnecessary iterations. In most cases, this simply ends-up generating source-trees - the tree generation strongly depends on the choice of the initial path and edges.
- The VM then uses *Simple Network Management Protocol (SNMP)* [50] to provide the different spanning-tree information to the switches.
- The VM continuously monitors the load characteristics of each pair of nodes, based upon the information provided by the VNC. When there is a significant load change, the VM triggers a topology reconfiguration.
- In the event of failures, the switches notify the VM (by means of SNMP traps). The VM checks which paths are affected and notifies the source hosts of the affected paths to switch to the available backup path. After this, the VM also triggers topology reconfiguration.

By having traffic monitored on end-hosts, Viking achieves the main advantage of optimizing the paths between every pair of nodes in terms of load balancing across the whole network. This provides good bandwidth efficiency and prevents one of the main problems with spanning-tree approaches, i.e., traffic concentration on critical links. Another advantage of Viking is the automatic computation of both primary and backup paths. This results in a very fast convergence (order of 400 to 600 milliseconds) given that traffic can be automatically diverted to the backup path, without the need to freeze the topology.

While interesting, the placement of components in the end-hosts is in fact one of the main weaknesses considering MAN scenarios. Not only does this implies changes to all the end-hosts, but it also implies that a central manager has to communicate with every single switch, as well as with the end-hosts. Furthermore, the performance of Viking is highly dependent on the fact that there should be two disjoint paths between every pair of end-hosts, and that the different spanning-trees should minimally overlap, fact which is strongly dependent upon the type of topology in use.

V. Alternative Ethernet Forwarding Approaches

This section provides an overview of approaches that attempt at improving the Ethernet forwarding by following directions alternative to the current IEEE spanning-tree (or extensions). For instance, some approaches follow a shortest-path direction, while others opt by a better-than-spanning-tree direction. What these approaches have in common is that they still keep the appealing and flexible connectionless nature of Ethernet.

A. Smartbridge

Smartbridge [51] was originally developed to be applied between different LANs , i.e., *inter-LAN*. The reason to develop such a solution relates to the fact that inter-LAN links are the ones that carry more traffic, and thus, the ones where bottlenecks may arise most. Smartbridge aims at improving inter-LAN performance by keeping the good properties of spanning-tree based approaches, while providing shortest-paths between every single pair of nodes.

To obtain topology knowledge, Smartbridge relies on *diffusing computation* [52]. In diffusing computation, an initiator sends a topology request to all of its neighbors, which then send the request to their neighbors, and so on. To confirm the whole process a reply is sent after each request. From the perspective of each diffused computation initiator, this method provides the means to know exactly when the whole distributed computation has finished. However, it does not prevent the creation of *transient* loops on the network. To prevent the creation of such loops, Smartbridge adds a method that ensures *effective global consistency,* i.e., Smartbridge relies on a mechanism that prevents a process from mixing old and new information.

Every time there is a topology change from the perspective of a Smartbridge, this bridge triggers a diffusing computation process that propagates to every bridge, collects the new topology in the form of a list which describes the connections between the different bridges and segments and then redistributes this list to the remainder bridges. Each instance of the topology acquisition is uniquely identified by a bridge identifier - corresponding to the identifier of the bridge that initiated the process - and an *epoch* number. The epoch number gives the means to know which of the topology acquisition instances is the one that a bridge should rely upon, given that there maybe concurrent instances running. The most recent acquisition that runs to completion contains the up-to-date topology[6]. During topology acquisition, which is in the order of tens of milliseconds, the information is *freezed* - packet dropping occurs.

This implies that for each segment on the network, a Smartbridge holds a port, *designated port,* which assigns a global identifier to the segment and which keeps information about the ports connected to the segment. It is by means of this port that announcements and reports on the membership status of hosts associated to the segment is provided, thus speeding up the convergence process.

After convergence of the topology view, Smartbridge nodes can perform traffic forwarding. For that, each Smartbridge holds a table containing the association between host MAC address and segment, which is provided by the designated port of a segment, as mentioned before. In contrast, STP-based approaches keep a table which holds the association between MAC and port - the true device location is not really known.

Known traffic forwarding is performed on shortest-paths (number of hops) between the corresponding segments. After the host location learning process, each Smartbridge contains a table which links host location to a specific segment, i.e., to the bridge designated port. When a Smartbridge receives a packet whose destination is known, it simply consults its internal databases and forwards the packets on the previously computed shortest-path. If, instead, the destination is unknown (or is a multicast destination), then the bridge relies on a source *unrooted* spanning-tree. The tree is unrooted given that the root is represented by the segment where the source MAC address is connected to and not by a bridge. This means that in fact the packet flooding is performed by the first bridge after the segment where the known source is. Such bridge is named *network flood talker*.

In case a frame arrives to a bridge with unknown source MAC address (e.g., the source host may have moved) then the respective bridge may trigger a *location revision wavefront* process, which spreads into the network. A location revision request consist of a deterministic breadth-first traversal of the topology graph process started at a chosen bridge - bridge with the largest identifier in the network. Such process results in the creation of a minimum-depth spanning-tree, the *Location Revision Spanning-Tree (LRST)* which is used to find out the segment where the MAC address resides. Because the computation is deterministic and the distributed graphs are identical, each Smartbridge is able to separately achieve the same result.

The main differentiating factors in Smartbridge are:

- each selected route is a shortest-path;
- the union of all routes starting on a LAN form a source-tree, which allows the quick detection of hosts that moved;
- the union of all routes ending on a LAN form a sink-tree.

Smartbridge claims that the time to stabilize a topology change is in the order of 20 milliseconds[7]. The state kept per bridge is a function of the number of bridges, the number of end-hosts, the number of segments, and the average number of ports per

[6] This process was originally developed to be used in Autonet [53].

[7] Simulation results incorporated several topologies with a maximum of 12 bridges.

segment. These are quite low convergence times, due to the use of diffusing computation.

However, and specifically considering MAN requirements, Smartbridge incurs several disadvantages, being the first one the lack of backward compatibility to IEEE 802.1-style bridges. Secondly, Smartbridge does not consider the use of backup paths, which implies that there may be still heavy packet loss during reconfiguration. Thirdly, not only does each bridge have to keep a table holding a full topology perspective, but it also has to keep the host MAC-to-segment association table (*host location table*) which in fact is claimed to consume the most of the Smartbridge available storage.

B. STAR: A Transparent Spanning-tree Bridge Protocol with Alternate Routing

Transparent Spanning-Tree Bridge Protocol with Alternate Routing (STAR) [54] is an approach that relies on enhanced bridges (STAR bridges) backward compatible with standard bridges. STAR specifically aims at enhancing the forwarding path performance while at the same time keeping backward compatibility. For that, STAR relies on the computation of "best effort" shortest-paths, in the sense that not all paths chosen are the shortest, but when available, STAR will give preference to such type of paths. This means that a path between s and d is always "shorter" than the regular spanning-tree path, where different metrics can be applied (e.g., delay). To achieve this, STAR relies on the computation of the full topology graph where links between STAR bridges that would be chosen as inactive for a spanning-tree computation can be re-used.

STAR bridges start by computing a spanning-tree covering all the bridges, i.e., STAR and legacy bridges. Then, path computation process is triggered. Before this process ends, STAR bridges together with legacy can perform regular learning and forwarding on the common spanning-tree. However, as soon as the path finding process ends, then STAR bridges switch to their own learning and forwarding processes. The end of the path finding process is announced by means of a timeout mechanism.

STAR bridges keep two different forwarding tables: *Bridge Forwarding (BF)* and *Host Location (HL)* tables. BF tables keep the best association between STAR bridges and ports in the form of Distance Vectors (DVs) per bridge, while HL tables keep the association between host MAC addresses and the respective STAR bridge, named *agent bridge*. Each BF DV contains:

- an estimated distance between the current bridge and the destination bridges;
- the forwarding port of the current bridge to reach the required destination bridge;
- the next-hop STAR bridge on the path to the destination bridge;
- a flag indicating whether the estimated distance is accurate;
- a flag indicating whether the path is an active tree path;
- a flag indicating the relation between the two bridges, i.e., ancestor, descendant, or otherwise.

The distance estimation between two bridges is based upon the difference of the distance between each bridge and the root of the original spanning-tree. Such information (and the remainder information kept in each DV entry) is obtained by means of two different STAR frames: *DV-MyInfo* and *DV-OurInfo*. DV-MyInfo frames carry information about a bridge's topology perspective, namely, distance to the root of the original spanning-tree, as well as information on the parent of the bridge, to be propagated to the other STAR bridges. DV-OurInfo frames carry information related to both a source and a destination STAR bridge. These two messages allow each STAR to compute the distance between two bridges but not all path distances can be accurately computed, particularly when the distance computation between two STAR bridges involves information between a third, not directly connected STAR bridge. Consequently, the STAR algorithm takes this into consideration - for the case where a tree path exists, it only considers additional paths whose distance could be accurately estimated.

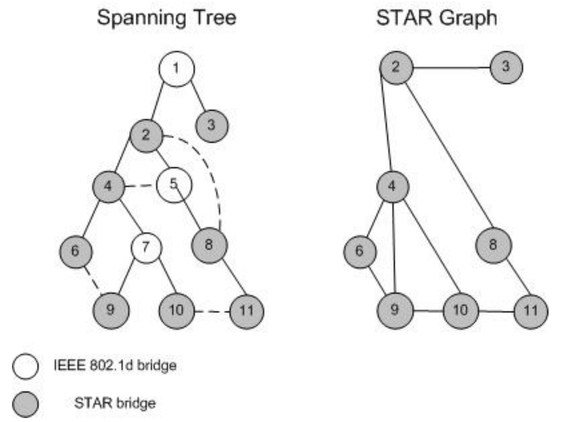


Fig. 11. STAR graph vs. spanning-tree.

When a data frame is received by a STAR bridge x, then x looks the MAC destination address d in its HL table. If the address is found, then x obtains the association between host MAC d and agent bridge MAC address. With the agent bridge MAC, x checks the BF database to determine the path to use. If, however, the agent bridge address is not found then x uses the regular FD and relies on the regular spanning-tree procedure.

Assuming that the host MAC destination address d is not known, then bridge x performs a regular broadcast procedure.

When a STAR bridge gets a frame from an unknown host (host whose MAC address has not been learnt yet), it starts by declaring itself as agent for the host and by sending a *HostLoc* frame to all other STAR bridges.

A host location is considered *known* when STAR bridges hold both the host MAC associated with the agent bridge, i.e., when the agent bridge for a specific host is known. Only agent bridges are allowed to forward data frame on the enhanced data paths. This avoids information inconsistency,

The ability to perform a DV exchange gives STAR bridges more flexibility to choose on which paths to forward traffic. Consequently, STAR bridges optimize the forwarding in the sense that path computation results in better than spanning-tree paths and hence, STAR can use links that otherwise would be blocked. Furthermore, by reducing path length STAR improves

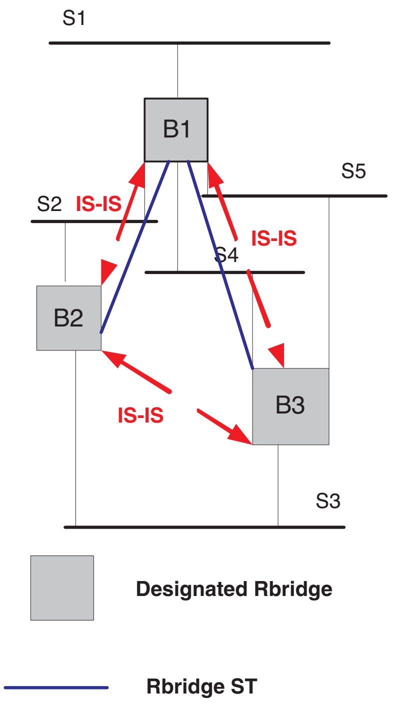


Fig. 12. Rbridge operation.

the overall latency and yet, keeps the backward compatibility with IEEE 802.1-compliant bridges.

However, the STAR exchange of information and graph creation is complex, which seems to hint that for large scale networks the described mechanism would not scale. Furthermore, there is no data related to resource utilization and to convergence times, given that the authors focused the results on message complexity, storage and overall performance, as well as compatibility with IEEE 802.1-style bridges.

C. RBridges

Rbridges [55], [56] is currently being defined by the IETF TRILL working-group and corresponds to a hybrid bridge concept, where bridges are enhanced to perform both Layer 2 and Layer 3 forwarding. The main purpose of these bridges is to glue together different physical segments (coupled by different bridges), so that they look like a single subnet for IP and includes, for now and with that purpose, some possible optimizations:

- *Address Resolution Protocol (ARP)* [57]/*Neighbor Discovery (ND)* [31] changes required to avoid the use of flooding in every situation;
- support secure neighbor discovery;
- hop-count (Time-to-Live (TTL)) for robustness when encountering temporary loops;
- no delay for hosts attached to the network;
- multicast support;
- be as secure as current bridges;
- define Layer 3 functionality to interconnect with Layer 2 functionality.

In order to learn local MAC addresses, Rbridges rely on regular MAC learning. Then, among themselves and as illustrated in Fig. 12, the new bridges rely on IS-IS [8] to exchange information concerning the learnt MACs and path (e.g., link costs). By using a link-state approach to flood information, this concept allows a quick distribution of locally learnt MACs without recurring to native broadcasts. Furthermore, relying on link-state routing to perform path computation gives Rbridges the means to perform shortest-path forwarding with multiple path and yet, prevent information duplication.

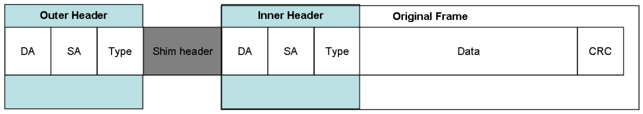


Fig. 13. Rbridges encapsulated frame format.

Rbridges rely on encapsulation to allow backward compatibility with current bridges. Encapsulation in Rbridges means adding both an outer MAC header and a shim header only used by Rbridges, as illustrated in Fig. 13.

The outer encapsulation header is a regular MAC header where the source address the MAC address of the transmitting Rbridge (changed on each Rbridge hop), while the destination address corresponds to the next-hop Rbridge (changed on each Rbridges hop). The shim header contains varied information depending on the type of situation. Normally and for unicast scenarios it must contain at least the MAC address of the egress Rbridge and a TTL hop-count which is used in Rbridges to deal with temporary loops and decremented on each hop, similarly to what happens in the IP layer.

The encapsulated frames are transmitted only between Rbridges. It is up to the destination Rbridge to de-encapsulate the original frame and to forward it to the required destination. This means that Rbridges form among themselves a LAN. Bridges learn the addresses of Rbridges by the regular spanning-tree method.

Concerning the different segments interconnected by Rbridges, only one Rbridges per segment, named *Designated Rbridge (DR),* is in charge of learning the identity of attached nodes, of flooding ARP/MLD requests when an ARP/MLD query to an unknown address is received, and of answering ARP/MLD queries.

As mentioned, Rbridges learn MAC/IP addresses on their subnet through ARP/MLD, as do regular bridges. Each Rbridge exchanges, by means of IS-IS, the learnt MAC/IP addresses associated with hosts attached to the segment they are responsible for. Therefore, all the Rbridges hold, for a fixed period of time, not only information about the host MAC addresses but also IP, and onto which bridges they are attached to.

This information holds the *soft-state* property, meaning that the learnt information can be re-used, without the need to perform more flooding. This means that Rbridges are capable of, not only forward Layer 2 packets, but also IP datagrams.

The forwarding of traffic to known (learnt) destinations is performed by means of the information learnt by the routing protocol. The forwarding of Layer 2 traffic to unknown destinations (either not learnt, or multicast) is performed

[8] Current choice goes to IS-IS, due to its ability to perform over Layer 2 directly. However, other link-state approach could be considered, e.g., OSPF.

over a regular spanning-tree that interconnects the different Rbridges. The IP forwarding is regularly processed by the routing protocol (e.g., IS-IS) learnt information.

Of course, as soon as multiple paths are supported, temporary loops may occur. For instance, in terms of unicast scenarios, it may happen that a new bridge is added to the topology merging two links. In this case, it can happen that two different Rbridges become responsible for the same segment and therefore, it might be hard to distinguish original frames from de-encapsulated ones. For this case, the hop-count cannot help, given that de-encapsulated frames discard this info. Consequently, Rbridges does not allow two bridges to become responsible for the same segment. The second situation happens for multicast traffic, which is regularly flooded over the Rbridge spanning-tree. This tree might have temporary loops and while for unicast routing the packets won't proliferate, for the spanning-tree case packets get duplicated. For this scenario, the Rbridges use a hop-count: the Rbridge that injects a packet into the spanning-tree can compute a minimal hop-count. The hop-count is therefore key to prevent the proliferation of information when loops occur. But additionally, other measures can be considered, to prevent the possible limited proliferation of information. Rbridges propose a timer, similar to the cache timer of regular bridges.

When compared to the current 802.1-style bridging, Rbridges brings in some advantages. For one, packets travel via shortest-path and multiple-path support is provided, being packet proliferation during transient loops controlled by means of proposed Rbridge. Given that transient loops are not a problem, topology changes can be sped up, based on local information.

Nonetheless, relying on a link-state protocol to perform information exchange may result in convergence problems, given that the key to an adequate convergence is adequate IS-IS timer tuning [58]. Systems that rely on link state protocols will also have to tolerate inconsistency intervals, while the protocol is converging. While the TTL added to Rbridges will help in avoiding exponential propagation, it will not help in optimizing the convergence times. Additionally, it would be necessary to announce the end of the reconfiguration process, given that link-state protocols provide no indication of termination. Plus, Rbridges flood all the (locally) learnt MAC addresses independently of those addresses being requested by a specific node. In other words, it may happen that the amount of flooded MACs to a specific Rbridge will never be used given that no end-hosts require such addresses as destinations. While such entries will eventually vanish by means of the inherent aging process associated with each entry of the forwarding table, this impacts negatively the performance of Rbridges. While these are issues that will have to considered if Rbridges would be applied to a MAC environment, it should be noticed that Rbridges are considered to be applied to the scope of LANs, e.g., a campus area.

VI. Connection-oriented Ethernet approaches

In an attempt to provide Ethernet transport with high level of reliability, manageability, and scalability, some approaches rely on Ethernet tunneling to build carrier-grade Ethernet services. The current main approaches are *Provider Backbone Bridging Support for Traffic Engineering (PBB-TE) [59] [60]*, *Transport MPLS (T-MPLS)* [61], and *VLAN Cross-Connect (VXC)* [62].

A. PBB-TE

Originally known as *Provider Backbone Transport (PBT)*, PBB-TE is a subset of the IEEE standard *Provider Backbone Bridging (PBB)* [63] (also known as MiM) currently in discussion in the IEEE.

From a high-level perspective, PBB-TE relies on point-to-point tunneling to establish services across the core of the MAC, disabling regular Ethernet features such as learning, flooding/broadcasting and the basic spanning-tree forwarding. The configuration of all the PBT connections is performed by means of a centralized (external) method.

The building blocks of PBB-TE are Ethernet nodes placed on the edges of a network and belonging to a specific provider. These nodes,*Provider Bridges (PB)*, are bridges that implement the Ethernet 802.1ad [26], an amendment of 802.1q. Among the different PBs, Ethernet tunnels are established to exchange information relying on the 802.1ad frame format.

Fig. 14 illustrates the three different frame types, namely, the original 802.1q, the QiQ format (802.1ad), and the PBB frame format (802.1ah). As described before, the original 802.1q frame holds a Q-tag which allows any operator to allocate 4094 VIDs and to mark each frame with a fixed priority scheme (802.1p, 3 bits).

The QiQ frame outer tag can be used to carry information concerning a provider, while the inner relates to customer information. This allows each customer to use the full space of the VID tagging scheme, i.e. 4K tags: the customer tags are "hidden" from the bridges in the core. On its side, the provider can configure. up to 4K S-VLAN supporting up to 4K customers each.

Another major difference between the 802.1ad format and the 802.1q is that the former holds, in place of the regular 802.1p static priorities, a more complete resource reservation management, based upon the re-interpretation of this field together with the *Canonical Format Indicator (CFI)*. The 3 bits used for 802.1p now hold a *Priority Codepoint (PCP)* field followed by a *Drop Eligible (DE)* field. Two tables are then established per port and relate to the encoding and decoding. The *PCP Encodingtable* holds 16 entries resulting from the combination of the 8 possible values of PCP and 2 of DE. The *PCP Decodingtable* holds 8 entries, corresponding to each of the possible PCP values. This provides more flexibility than the fixed format allowed by 802.1p.

While promising in terms of increased scalability, PB does little to prevent the so-called *MAC address table explosion*: the bridges belonging to the provider still have to learn all the customer MAC addresses that frames carry. A solution to this problem is MiM encapsulation, of which PBB is the most relevant example. As shown in Fig. 14, the PBB frame (802.1ah) re-engineers the Q-tags now as *B-tag* (*Backbone tag*) and as *I-Tag* (*Service Instance Tag*). The I-Tag contains information representing a logical service instance, which

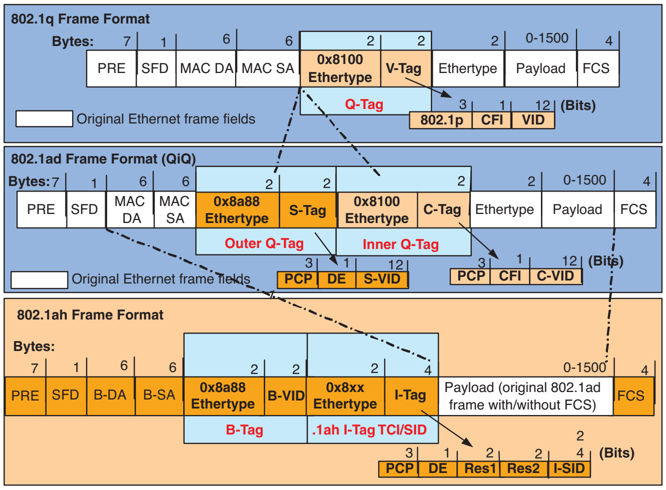


Fig. 14. Frame formats for 802.1q, 802.1ad (PB), 802.1ah (PBB).

allows to associate specific features of the service to customer records on the edge devices, while the B-Tag together with the MAC address of the egress PB represents the provider tunnel. In addition, PBB adds a second MAC header to the frame, being the original 802.1ad frame carried as payload. This means that the bridges in the core are not aware (and hence do not keep state) concerning the customer information. The ingress PB prepends an outer MAC header which only relates to the core switches: *Backbone Source Address (B-SA)* corresponds to the MAC address of the ingress PB, while *Backbone Destination Address (B-DA)* represents the MAC address of the egress PB. The B-Tag field represents the backbone tunnel identifier, while the I-Tag corresponds to an *Instance Tag*. Once the frame reaches the egress PB, it gets decapsulated and forwarded to the original (customer) destination. While the new frame format gives the means for PBB to prevent MAC address explosion, PBB still relies on spanning-tree based forwarding (e.g., provided by RSTP) and consequently does nothing in terms of improving the bandwidth inefficiency resulting from links that are blocked in spanning-tree approaches.

Attempting at overcoming the open issues of PBB, PBT builds upon PBB by means of a sophisticated management platform which helps in provisioning the adequate set of paths with resilience and reliability. An example of the global PBB-TE operation is provided in Fig. 15, where a provider is serving two different customers 1 and 2. Customer 1 wants to interconnect sites A, B, C, while customer 2 wants sites D and E interconnected. Customer 1 traffic between the three sites is covered by a customer VLAN to which the VID tag 100 is assigned. Customer 2 also opted to label traffic to be transmitted between sites D and E by means of the same VID, 100. The provider then assigns traffic of customer 1 to the 24-bit I-SID 10000, while the traffic from customer B sites is assigned to an I-SID 4000. This configuration is provided in the PBT edge bridge X. In the case of the example provided, the two I-SIDs are mapped to two different primary tunnels. Notice however that each service instance (I-SID) can be mapped to both primary and backup tunnels, in order to increase the resilience of the provided tunnel infrastructure.

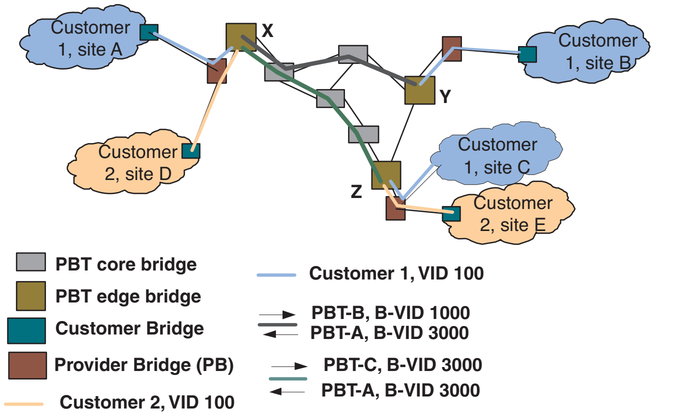


Fig. 15. PBB-TE operational example.

The tunnel that the provider established for customer 1 traffic from site A to B is identified by the B-VID 1000 together with the MAC address of the PBT bridge Y. Traffic that flows from customer 1 site B to site A carries in the header the B-VID 1000 together with the MAC address of the PBT bridge X. When traffic from site A destined to site B arrives to PBT X, this bridge adds the new encapsulation header holding as B-DA the MAC address of PBT Y, as B-SA the MAC address of PBT X, as B-VID the value of 1000 and as I-SID the chosen value 10000. The traffic is forwarded along the different PBT core bridges which perform regular MAC learning based on the outer header (customer information is hidden in the core). It should be noticed that the core bridges do not perform learning or flooding related to the assigned B-VID (1000), given that this VID has been reserved for PBT use. Consequently, each core bridge must be provisioned with adequate forwarding tables in order to be able to properly forward traffic in the established tunnels. In the given example, each of the core bridges along the path between site A and B must hold an entry in their forwarding tables for the tunnel (PBT Y, B-VID 1000). When PBT Y gets the encapsulated traffic, it realizes that the *Instance Service Identifier (I-SID)* 10000 is actually associated to the Service VLAN Identifier (S-VID) 1000 and consequently, forwards the original customer traffic to the next PB. In the same way, traffic of customer 1 being exchanged between sites A and C is mapped in PBT X to the tunnel identified by (PBT-Z, Backbone VLAN Identifier (B-VID) 3000).

The tunneling and required information to properly forward data must be pre-configured by means of a management system. Customer information, as well as services must be associated with tunnels which can be monitored by means of IEEE *802.1ag Connectivity Fault Management (CFM) Continuity Check Messages (CCM)* [64]. CCM controls how frames are transmitted/received across the established tunnels; in case a primary tunnel fails, then CCM allows the endpoints to activate the backup tunnel (which must also be pre-configured in the forwarding tables of all the PBT bridges).

While interesting, it is clear that the major weakness of PBB-TE relies on the complexity inherent to the configuration in every single equipment that participates in a specific topology/service. It may easily happen, for instance, that problems

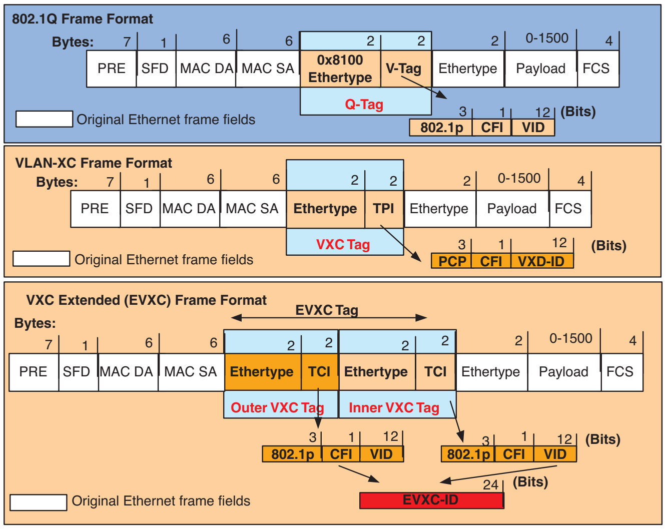
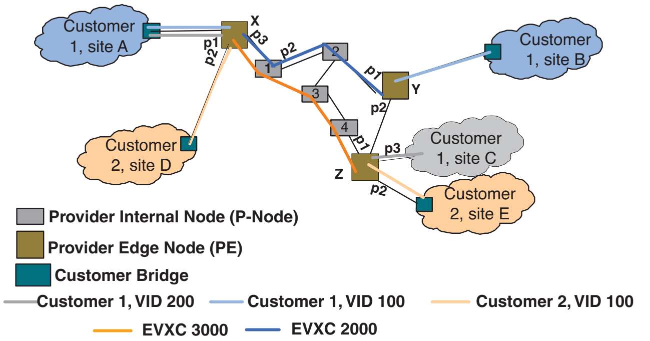


Fig. 16. VLAN-XC frame formats compared to the QiQ frame format.



PE A

Ingress Port	Ingress VLAN	Egress Port	Egress VLAN
p1	100	p3	2000
p2	100	p3	3000
p1	200	p3	3000

PE B

Ingress Port	Ingress VLAN	Egress Port	Egress VLAN
p1	2000	p3	100

P-Node 1

Ingress Port	Ingress VLAN	Egress Port	Egress VLAN
p1	2000	p2	2000
p1	3000	p2	3000

PE C

Ingress Port	Ingress VLAN	Egress Port	Egress VLAN
p1	3000	p3	200
p1	2000	p2	100

Fig. 17. VXC operation example.

result from configuration or operator mistakes. Within IEEE 802.1 there is currently an attempt at providing solutions related to management errors [65].

B. VLAN Cross-Connect

VXC [62][9], also known as *Provider VLAN Transport (PVT)*, is an approach which relies on a re-engineering of the Q-Tag to perform a switching scheme similar to the one of ATM: instead of the regular forwarding in VLANs where entries associate MAC addresses to a specific VID, VXC redraws the Q-Tag (named *VXC tag*) and perform forwarding based on the ingress port and the VXC tag, completely independent of the destination MAC addresses. This approach also solves the MAC table address explosion given that switches along the path don't need to associate MACs to ports. The frame format used by VXC elements and which is illustrated in Fig. 16 is based upon the original 802.1q format, where the Q-tag has been re-engineered in a way that allows VXC to co-exist with legacy equipment. In other words, the VXC space is split into legacy and VXC VID and consequently, VXC elements know how to treat the marked frames received. Being directly based in 802.1q, this format inherits the 802.1q scalability problems. However, while on 802.1q the VID limitation of 12 bits is from an end-to-end perspective (4K VLANs per provider), for VXC this limitation occurs *per* port (4K VLANs per port).

The *Extended VLAN Cross-Connect (EVXC)* frame format provides a clever solution to the scalability problem. EVXC is based upon QiQ and concatenates the VID of each Q-Tag, thus supporting a resulting VID of 24bits. This allows a provider to map 16M VIDs per port.To provide an example on how VXC operates, Fig. 17 relies on the scenario already described in Fig. 15, where two different customers 1 and 2 wanting to interconnect different sites: customer 1 wants to cover sites A, B, and C, while customer 2 wants sites D and E interconnected. As illustrated, PE nodes placed at the borders of the provider domain are responsible for triggering/terminating VXC connections, while P-Nodes simply rely on the VXC information carried in frames to perform adequate forwarding.

Between sites A and B, the ingress PE X takes care of adequately mapping the different ingress VLAN to the adequate egress (VXCspace) VLANs. It should be noticed that the mapping is now based upon ingress port and VIDs. In contrast, in regular VLAN bridging the mapping is done by means of MAC addresses associated to VIDs. At each hop along the path, the mapping between ingress ports and VLANs is again established. For instance, traffic of customer 1 being exchanged between sites A (in bridge PE X this corresponds to the mapping <p1, 100> and Y is mapped in PE X to VID 2000, while traffic of customer 2 being exchanged between sites D and E (in PE X, this is identified by <p2, 100>) is mapped to VID 3000. A remark should be done for the case of traffic of customer 1 being exchanged between sites A and C: here, and in contrast to both regular VLAN bridging and PBB-TE, it would be necessary to have a second customer VLAN (VID 200) covering the traffic of both sites: currently, VXC does not seem to support point-to-multipoint (nor multipoint-to-multipoint) scenarios.

VXC requires, as PBB-TE, adequate configuration along the path. The control plane of VXC is still left open but strong emphasis is being put in the *Generalized Multiprotocol Label Switching (GMPLS)* [67] as the main candidate for such support. Furthermore, the IEEE raised concerns in terms of the 24-bit tag, claiming that it is an Ethernet architectural violation, which suggests that PVT is unlikely to emerge very soon as a potential competitor in terms of standardization.

C. T-MPLS

T-MPLS [68], [61], can be seen as an MPLS derivate whose application field is Layer 2. In contrast, MPLS incorporates both Layer 2 and Layer 3 functionality support. Having been stripped from the Layer 3 (connectionless) functionality, the intent behind T-MPLS is to rely on a specific subset of MPLS sufficient to provide connection-oriented packet transport.

[9] VXC is also being discussed in the context of the IETF working group GELS [66] and in the ITU-T ST 15.

TABLE II
ITU-T RECOMMENDATIONS FOR T-MPLS.

Recommendation	Title
G.8110.1	Architecture of Transport MPLS (T-MPLS) Layer Network
G.8112	Interfaces for the Transport MPLS (T-MPLS) Hierarchy
Y.17tom	Operation & Maintenance mechanisms for T-MPLS layer networks
G.8121	Characteristics of Transport MPLS equipment functional blocks
G.8131	Linear Protection switching for Transport MPLS (T-MPLS) networks
G.8132	Shared Protection Ring for T-MPLS networks

While most of the focus is on Ethernet services, T-MPLS is claimed to support all packet services on top of SDH circuit switches.

T-MPLS functionality (cf. Table II) falls under the umbrella of the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) (in contrast to the MPLS functionality which falls into the umbrella of the IETF) and consequently its background is based upon the traditional carrier transport network values which include tight control, lowest cost-per-bit over service, and simple service aggregation. Nonetheless, and because MPLS falls into the umbrella of the IETF, there has been lack of consense between the IETF and the ITU-T perspective. This situation culminated with the ITU-T attempting to reserve a range of MPLS labels for its own use, which would infringe the ownership boundary of the IETF in MPLS. In July 2006 a joint (ITU-T interim) meeting took place and a revised recommendation for T-MPLS [68] has been generated.

The T-MPLS is placed in Layer 2 underneath MPLS (or IP/MPLS) and includes the following main features:

- **MPLS forwarding** behavior stripped down of IP functionality.
- **No integrated OAM and survivability** - these aspects are left to the transport network.
- **No integrated control plane**, nor any recommendation to follow a specific approach [10].
- **No label reservation**. Labels in use by T-MPLS are not reserved for its own use; instead, they come from the MPLS space and special labeling requirements must be coordinated with the IETF and MPLS standards in order to ensure interoperability.
- **Support for bidirectional LSPs**. T-MPLS ties together unidirectional LSPs between a specific pair of nodes, being the state concerning the pairing association kept in each node along the LSP path.
- **No Equal-Cost Multiple Path (ECMP) support**. The reason to remove ECMP is the claim that in a connection-oriented (optical world) load-balancing is not needed, given that traffic can follow two paths with equal cost.
- **No Penultimate Hop Popping (PHP) support**. PHP is a feature that allows labels to be removed one node before the egress node in order to reduce the required processing power on the egress node. It has been removed given that it is incompatible with Y.1711 OAM.
- **No LSP merge support**. LSP merge allows the traffic forwarded by the same path (sharing links) to the same destination to rely on a single label. While LSP merging increases scalability, it cannot be used in a connection-oriented concept, given that it hides the source information.
- **Simplification of discard algorithms**. For applications requiring some form of loss probability, T-MPLS requires on a single drop precedence. In applications that require statistical multiplexing gain, only two drop precedence values are supported.

The fact that T-MPLS specifications must cope with the views of two different standardization entities (IETF and ITU-T) makes it more difficult to achieve a stable version of the solution. As of today, there are still some items under discussion, namely:

- **Control plane**. This is one of the major concerns of the IETF, who considers that simply stating that the control plane is null jeopardizes the proposal and goes against the MPLS principles. The preference for the use of *Automatically Switched Optical Network (ASON)*/GMPLS, who has as an ambitious goal to become a global control plane for all networks is likely to increase the complexity to the approach and delay the ongoing debate. Consequently, the provisioning may remain manual for a long time.
- **OAM**. The Y.1711 OAM (now part of T-MPLS) is not part of MPLS, which is one of the points leading to incompatibility management-wise between the two approaches.
- **Interoperability**. Interworking between IP/MPLS and T-MPLS pseudowires is still in an early stage and requires close cooperation between both entities.
- **Client/server architecture**. The basis of T-MPLS is that it can carry any type of packet-based service, including IP, MPLS, and even other instances of T-MPLS. In contrast, IP/MPLS operation is done in a flat or peering model. Consequently, this is another issue affecting the global interoperability that is claimed to be the universal property of T-MPLS.

D. Connection-Oriented Approaches in a Nutshell

As described throughout the previous section, today there is strong support from the traditional incumbent-oriented telecommunication perspective to pursue Ethernet services in a connection-oriented manner. The main approaches under discussion in the different standardization and technology fora are PBT, VXC, and T-MPLS.

The mentioned connection-oriented approaches have some points in common: all of them are being careful in terms of changes to Ethernet in an attempt to support backward-compatibility, or at least to allow a fall-back to IEEE 802.1 bridging in some scenarios. All of them support packet prioritization and dropping, as well as some type of protection.

PBB-TE incorporates some promising traffic-engineering mechanisms (MiM, double tagging) to allow a complete customer/provider separation. In terms of forwarding in the backbone, it relies on a 60-bit address (B-VID with 12 bits; MAC of egress PE, 48 bits) to uniquely identify a backbone pipe. Given that PBT considers a unique source MAC, OAM traceback is simplified. MAC learning has been disabled, being the Ethernet tables populated by some form of external

[10] GMPLS is currently the most popular control approach being cited for any connection-oriented Ethernet solution, T-MPLS included.

TABLE III
BASIC FEATURE COMPARISON BETWEEN ETHERNET (VLAN BRIDGING) PBB-TE, VXC, T-MPLS.

Solution	**Tunneling/Service Forwarding**	**Technology**	**QoS**	**Protection**	**Traffic aggregation**	**OAM**	**Supported connections**			**Standardisation status**
							1:1	**1:N**	**N:N**	
Ethernet (VLAN bridging)	MAC+VLAN	Connectionless; address resolution based on MAC learning/broadcasts; STPapproaches for loop avoidance; VLAN configuration performed manually (no integrated control/management plane)	802.1p	1:1	16 M VLANs per provider network	802.1ag, manual	**Yes**	**Yes**	**Yes**	Standardised
PBB-TE	Tunnel: MAC+B-VID; Service: I-SID	Connection-oriented; supports both raw Ethernet or 802.1ad; address resolution only required on the edges (no customer MAC learning in the core); requires sophisticated management plane for tunnel creation and bandwidth management	PCP/DE	1:1; 802.1ag monitoring; ITU-T G.8031 – SG 15 Ethernet protection	16M VLAN per tunnel (60-bit tunnel identifier)	802.1ag, ITU-T Y.1731 - SG 13 Ethernet OAM, manual	Yes	Yes	Yes	PBB as draft (802.1ah); PBB_TE under discussion
VXC	Ingress Port+VLAN	Connection-oriented, similar to ATM switching; re-engineering of QiQ; MAC learning only at the edges (no customer MAC learning in the core); requires sophisticated management plane for tunnel creation and bandwidth management on a hop-by-hop basis	802.1p	1:1	16M VLAN per port	Manual	Yes	No	No	Early discussion, IEEE, IETF, ITU-T
T-MPLS	Tunnel: MPLS LSPs; Service: PW/VPLS label at bottom of stack	Connection-oriented; subset of MPLS (no IP functionality); specifically designed for packet-based service support by means of point-to-point connections; requires sophisticated management plane for tunnel creation and bandwidth management	CS tags	1:1; possibly fast rerouting; ITU-T recommended Y.1720/G.813 linear protection switching	20-bit label	ITu-T Y.1711 (ITU-T OAM mechanisms for MPLS)	**Yes**	**Yes**	**No**	ITU-T Work in progress (SG 15)

configuration or signaling (e.g., GMPLS signaling). For active sessions, PBB-TE relies on Y.1731 OAM to manage the active connections. Finally, the backward compatibility (and coexistence) is achieved by partitioning the VID space with Ethernet bridging.

VXC relies instead on a new interpretation of the QiQ format, together with forwarding based upon the association of ingress ports to VLANs. To achieve scalability (because the VID space is of 12 bits), VXC swaps (similar to swapping in ATM and MPLS) the VIDs on each hop along the path, which makes the VID locally significant only. VXC even goes further in terms of scalability, proposing an extended VID format (EVXC) which is based on the interpretation of a concatenated VIDs of the two Q-tags. This then allows to have 16M VIDs per port on each device. Similarly to PBB-TE, VXC connections may also be managed by means of Y.1731 OAM, and backward compatibility is ensured by partitioning the VID space between VXC and non-VXC.

T-MPLS treats Ethernet services as "client" in an MPLS connection-oriented network. The labels only have significance locally and are used in the core instead of MAC addressing. Y.1711 OAM also seems to be the choice in terms of connection management. Finally, backward compatibility is not possible in what concerns MPLS, even though interworking mechanisms are being considered.

Given the current state of evolution of the three approaches, it is still unclear which may lead in terms of Ethernet service support. It is clear that in terms of backward compatibility and Ethernet services being defined by the MEF, PBB-TE leads the run. On the other hand, T-MPLS goes ahead in terms of standardization. Nonetheless, there are open issues that must be solved in any of the approaches. From those, two should be highlighted: none of the three approaches really provides adequate support to multipoint-to-multipoint services and none of them really has a clear saying in terms of an adequate control plane, which is crucial to the proper provisioning of services with any of the approaches, given that all three require some form of sophisticated configuration to properly establish the logical topologies that are the basis to carry Ethernet services.

Table III provides a summary of the main characteristics for each of the three approaches. Related work summingly analyzing these approaches can be found in [69].

VII. SUMMARY AND CONCLUSIONS

This document provides a survey of past and current directions in what concerns a series of work which focus in enhancing several aspects of Ethernet forwarding in a way that allows Ethernet to become applicable to large scale networks of which the MAN is an example.

The survey starts by providing Metro Ethernet and MAN notions, as well as the main Metro Ethernet services currently being dictated by different standardization organizations. The document then describes the different IEEE spanning-tree approaches, given that these are the basis for the current Ethernet forwarding. An overview is next provided on approaches that still rely on some sort of spanning-tree based forwarding but that enhance some aspects of it, such as convergence times. Then, the survey goes over novel approaches that aim at leveraging the Ethernet forwarding with optimal path (shortest-path) and multiple path support while leaving its data plane intact. The direction of "connection-oriented" Ethernet is also described, by explaining the basics behind the most promising connection-oriented approaches, as well as advantages and disadvantages.

As described, while Ethernet is a promising technology, there are several issues that prevent it from adequately support transport services in the MAN. The different works (and different directions) described are the realization of this problem.

The connectionless approaches allow to take advantage of the full Ethernet potential (plug&play, flexible) and create room to easily deploy any type of service (multipoint-to-multipoint) from a data plane perspective. Nonetheless, by adding the flexibility to have multiple paths and to dynamically perform path computation, it is also necessary to consider an adequate control plane which current Ethernet standards do not support. Without adequate configuration and an adequate control plane, it is unlikely that the approaches mentioned can scale to large networks, as required in the MAN. Furthermore, one of the problems with the current approaches is that while most of them do provide one form or another for loop mitigation, none of them seriously addresses the MAC address explosion problem space.

Connection-oriented approaches have been emerging mostly from a telecommunication's vendor (and operator) perspective and as such, their positioning in terms of standardization is clearer than connectionless approaches. Nonetheless, the current given support results mostly from the fact that connection-oriented paradigms automatically provide a tighter control of path configuration. Tight control is not normally synonymous with cost-efficiency, and the current approaches being proposed will only prevail in large networks if a sophisticated management system is in place and capable of properly provisioning the required connections. But the main risk with current connection-oriented approaches is the fact that some Ethernet services which as of today represent a niche (such as multipoint-to-multipoint) are being treated as secondary priority. While today such services do fall in a market niche, it may happen that in the future they become a significant part of the transported services across a MAN (as starts to be the case for services such as IPTV) and this should not be disregarded when devising any novel Ethernet forwarding approach.

VIII. Acronyms

AAA Authentication, Authorisation, Accounting
AMSTP Alternative Multiple Spanning Tree Protocol
AN Access Node
ARP Address Resolution Protocol
ASON Automatically Switched Optical Network
ASP Application Service Provider
ATM Asynchronous Transfer Mode
B-DA Backbone Destination Address
BE Best Effort
BF Bridge Forwarding
BGP Border Gateway Protocol
BID Bridge Identifier
BPDU Bridge Protocol Data Unit
BRAS Broadband Remote Access Server
B-SA Backbone Source Address
B-Tag Backbone Tag
B-VID Backbone VLAN Identifier
CCM Continuity Check Messages
CFI Canonical Format Indicator
CFM Connectivity Fault Management
CIST Common Internal Spanning Tree
CP Customer Premises
CPE Customer Premises Equipment
CST Common Spanning Tree
C-VLAN Customer VLAN
DB Designated Bridge
DE Drop Eligible
DR Designated Rbridge
DSCP Differentiated Services Codepoint
DSL Digital Subscriber Line
DV Distance Vector
EAPS Ethernet Automatic Protection Switching
ECMP Equal-Cost Multiple Path
E-LAN Ethernet LAN
E-LINE Ethernet Line
EN Edge Node
EoA Ethernet over ATM
EoMPLS Ethernet over MPLS
EPON Ethernet Passive Optical Network
ER Edge Router
ERP Ethernet Ring Protection
E-TREE Ethernet Tree
EVC Ethernet Virtual Circuit
EVXC Extended VLAN Cross-Connect
FD Forwarding Database
GARP Generic Attribute Registration Protocol
GMPLS Generalized Multiprotocol Label Switching
GMRP GARP Multicast Registration Protocol
GOE Global Open Ethernet
GRE Global Routing Encapsulation
GVRP GARP VLAN Registration Protocol
HL Host Location
H-VPLS Hierarchical VPLS
IEEE Institute of Electrical and Electronical Engineers
IETF Internet Engineering Task Force
IGMP Independent Group Multicast Protocol
IP Internet Protocol
IPTV Internet Protocol TV
IPv4 IP version 4
IPv6 IP version 6
I-SID Instance Service Identifier
IS-IS Intermediate System-to-Intermediate System
ISP Internet Service Provider
IST Internal Spanning Tree
I-Tag Instance Tag
ITU-T International Telecommunication Union Telecommunication Standardization Sector
L2TP Layer 2 Tunneling Protocol
LAG Link Aggregation
LAN Local Area Network
LDP Label Distribution Protocol
LRST Location Revision Spanning-Tree
LSP Label Switching Path
MAC Media Access Control
MAN Metropolitan Area Network
ME Metro Ethernet
MEF Metro Ethernet Forum
MiM MAC-in-MAC
MLD Multicast Listener Discovery
MPLS Multi-Protocol Label Switching
MMRP Multicast Multiple Registration Protocol
MRP Multiple Registration Protocol
MVRP Multiple VLAN Registration Protocol
MST Multiple Spanning Tree Region
MSTI Multiple Spanning Tree Instance
MSTP Multiple Spanning Tree Protocol
MTU Multi-Tenant Unit
MVR Multicast VLAN Registration
NAP Network Access Provider
ND Neighbor Discovery
NIC Network Interface Card
NSP Network Service Provider
NT Network Terminator
OAM Operation, Administration, Maintenance
OUI Organizationally Unique Identifier
P2P Peer-to-Peer
PB Provider Bridges
PBB Provider Backbone Bridging
PBB-TE Provider Backbone Bridging Support for Traffic Engineering
PBT Provider Backbone Transport
PC Personal Computer
PCP Priority Codepoint
PD-MRSTP Per-Destination Multiple Rapid Spanning Tree

PDU Packet Data Unit
PE Provider Edge
PHP Penultimate Hop Popping
PIM-SM Protocol Independent Multicast-Sparse Mode
PIM-SSM Protocol Independent Multicast Source Specific Multicast
PPP Point-to-Point Protocol
PSN Packet Switched Network
PVC Permanent Virtual Circuit
PVST Per-VLAN Spanning Tree
PVT Provider VLAN Transport
PW Pseudowire
QiQ Q-in-Q
QoS Quality of Service
Rbridge Routing Bridge
Rbridges Route Bridges
RNP Regional Network Provider
RSTP Rapid Spanning Tree
SDH Synchronous Digital Hierarchy
SNMP Simple Network Management Protocol
SONET Synchronous Optical Networking
SP Service Provider
ST Spanning Tree
STAR Transparent Spanning-Tree Bridge Protocol with Alternate Routing
STB Set Top Box
STP Spanning Tree Protocol
S-VID Service VLAN Identifier
S-VLAN Service VLAN
TC Topology Change
TCN Topology Change Notification
TDM Time Division Multiplexing
T-MPLS Transport MPLS
TRILL Transparent Interconnection of Trillions of Links
TTL Time-to-Live
UE User Equipment
VC Virtual Circuit
VID VLAN Identifier
VLAN Virtual Local Area Network
VM Viking Manager
VMAN Virtual MAN
VNC Viking Network Controller
VPLS Virtual Pprivate LAN Service
VPN Virtual Private Network
VPWS Virtual Private Wire Service
VXC VLAN Cross-Connect

REFERENCES

[1] G. Chiruvolu, A. Ge, D. Cosaque, M. Ali, and J. Rouyer, "Issues and Approaches on Extending Ethernet Beyond LANs," *IEEE Commun. Mag., Special Issue on Ethernet transport over Wide Area Networks*, Mar. 2004.

[2] IEEE, "802.1D Standard for local and metropolitan area networks - Media Access Control (MAC) Bridges," June 2004.

[3] Siemens IC R&D WON and Pedro Nunes (Editor), "ERP Concept Specification," *Concept Paper*, Mar. 2004.

[4] Extreme Networks, "Ethernet Automatic Protection Switching (EAPS)," *White Paper*, 2002.

[5] IEEE, "Rapid Reconfiguration of Spanning Tree," *IEEE 802.1w (incorporated into IEEE 802.1D-2004)*, 2004.

[6] IEEE, "Multiple Spanning Trees," *IEEE Std 802.1s*.

[7] Telcordia, "GR253-Core: Synchronous Optical Network (SONET) Transport Systems," 2005.

[8] ITU-T, "Recommendation G.707: Network node interface for the synchronous digital hierarchy (SDH)," 2003.

[9] ITU-T, "Recommendation G.708: Synchronous digital hierarchy (SDH) network to network interface (NNI)," 2003.

[10] DSL Forum, "Broadband Remote Access Server (BRAS) Requirements Document," *DSL Forum TR-092*, Aug. 2004.

[11] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," *IETF RFC 3031*, January 2001.

[12] DSL Forum, "Architecture Requirements for the Delivery of Advanced Broadband Services," *DSL Forum 2003-427*, Nov. 2003.

[13] DSL Forum, "Migration to Ethernet Based DSL Aggregation," *Working Text WT-101*, Oct. 2004.

[14] DSL Forum, "Migration to Ethernet Based DSL Aggregation for Architecture and Transport Working Group," *DSL Forum WT-101, rev.3*, Oct. 2004.

[15] B. Cain, I. Kouvelas, B. Fenner, and A. Thyagarajan, "Internet Group Management Protocol version 3," *IETF RFC 3376*, Oct. 2003.

[16] D. C. magazine, "ISP backbones," September 1997.

[17] M. Batayneh, D. A. Schupke, M. Hoffmann, A. Kirstaedter, and B. Mukherjee, "On reliable and cost-efficient design of carrier-grade ethernet in a multi-line rate network under transmission range constrains," *Post Deadline paper, OFC'07*, March 2007.

[18] IEEE, "IEEE." Available at http://www.ieee.org/.

[19] Metro Ethernet Forum, "Metro Ethernet Networks - A Technical Overview," 2004.

[20] IETF, "L2 Virtual Private Networks (l2vpn) Working Group," *http://www.ietf.org/html.charters/l2vpn-charter.html*, Sept. 2004.

[21] IEEE, "IEEE 802.3ah - Ethernet in the First Mile," *IEEE Std 802.3ah-2004*, 2004.

[22] L. Anderson, P. Dolan, N. Feldman, A. Fredette, and B. Thomas, "LDP Specification," *IETF RFC 3036*, January 2001.

[23] L. Martini, L. Tappan, D. Vlachos, C. Liljenstolpe, G. Heron, and K. Kompella, "Transport of L2 Frames over MPLS," *Internet Draft (Expired)*, June 2004.

[24] K. Kompella and Y. Rehkter, "Virtual Private LAN Service," *IETF Internet Draft (Work in Progress)*, May 2004.

[25] IEEE, "Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks," *IEEE Std 802.1q*, 1998.

[26] IEEE, "802.1ad - Provider Bridges," tech. rep., IEEE, May 2006.

[27] I. Hadzic, "Hierarchical MAC address space in public Ethernet networks," *Globecom*, 2001.

[28] IEEE, "Guidelines for the Use of a 48-bit Global Identifier (EUI-48)," *http://standards.ieee.org/regauth/oui/tutorials/EUI48.html*.

[29] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei, "Protocol Independent Multicast Sparse Mode (PIM-SM): Protocol Specification," *IETF RFC 2362*, June 1998.

[30] S. Bhattacharvya, "An Overview of Source Specific Multicast," *IETF RFC 3569*, July 2003.

[31] R. Vida, L. Costa, R. Zara, S. Fdida, S. Deering, B. Fenner, I. Kouvelas, and B. Haberman, "Multicast Listener Discovery version 2 (MLDv2) for IPv6," *Internet Draft presentation, 50th IETF meeting*, Mar. 2001.

[32] Dell Computers, "Understanding IGMP Snooping," *White paper*, 2004.

[33] W. Fenner, "IGMP-based Multicast Forwarding ("IGMP Proxying")," *IETF Draft (expired)*.

[34] IEEE, "Part 3: Media Access Control Bridges," *IEEE Draft Standard 802.1d*, 1998.

[35] IEEE, "IEEE Standard 802.1ak for Local and Metropolitan Area Networks Virtual Bridged Local Area Networks - Amendment 07: Multiple Registration Protocol," June 2007.

[36] A. Myers, T. S. Eugene Ng, and H. Zhang, "Rethinking the Service Model: scaling Ethernet to a Million Nodes," *Hotnets III*, Nov. 2004.

[37] R. Pallos, J. Farkas, I. Moldovan, and C. Likovszki, "Performance Evaluation of the Rapid Spanning Tree Protocol in Access and Metro Networks," *IEEE AccessNets 2007, Ottawa, Canada.*, August 2007.

[38] K. Elmeleegy, A. L. Cox, and T. S. E. Ng, "On Count-to-Infinity Induced Forwarding Loops in Ethernet Networks," *IEEE Infocom 2006*, 2006.

[39] IEEE, "Multiple Spanning Trees," *IEEE Std 802.1s*.

[40] T. Cinkler, I. Moldovan, A. Kern, C. Lukovszki, G. Sallai, " Optimizing QoS aware ethernet spanning trees," *International Conference on Multimedia Services Access Networks, 2005. MSAN '05.* , June 2005.

[41] M. Padmaraj, S. Nair, M. Marchetti, G. Chiruvolu, M. Ali, "Traffic engineering in enterprise ethernet with multiple spanning tree regions," *Systems Communications Proceedings, 2005.*, vol. 14-17, pp. 261–266, August 2005.

[42] M. Padmaraj, S. Nair, M. Marchetti, G. Chiruvolu, M. Ali, A. Ge , "Metro Ethernet traffic engineering based on optimal multiple spanning trees," *Wireless and Optical Communications Networks, 2005. WOCN 2005. Second IFIP International Conference on Volume , Issue , 6-8 March 2005 Page(s): 568 - 572*, vol. 6-8 March 2005, pp. 568–572, 2005.

[43] A. Iwata, Y. Hidaka, M. Umayabashi, N. Enomoto, and A. Arutaki, "Global Open Ethernet (GOE) System and its Performance Evaluation," *IEEE J. Select. Areas Commun., Vol 22, number 8*, Oct. 2004.

[44] G. Ibanez and A. Azcorra, "Alternative Multiple Spanning Tree Protocol (AMSTP) for Optical Ethernet Backbones," *LCN'04*, Mar. 2004.

[45] IEEE, "Virtual Bridged Local Area Networks - Amendment 9: Shortest Path Bridging," *P802.1aq/D0,3 Draft Amendment to IEEE Std 802.1q-2005*, May 2006.

[46] IETF, "Transparent Interconnection of Lots of Links (TRILL) working group charter." Available at http://www.ietf.org/html.charters/trill-charter.html, 2006.

[47] D. Oran (Editor), “OSI IS-IS Intra-domain Routing Protocol,” *Request for Comments 1142, Internet Engineering Task Force*, Feb. 1990.

[48] S. Sharma, K. Gopalan, S. Nanda, and T. Chiueh, “Viking: A Multi-spanning Tree Ethernet Architecture for Metropolitan Area and Cluster Networks,” *Infocom*, 2004.

[49] Cisco Systems, “Per VLAN Spanning Tree (PVST),”

[50] J. Case, M. Fedor, M. Schoffstall, and J. Davin, “A Simple Network Management Protocol (SNMP),” *IETF RFC 1157*, May 1990.

[51] T. L. Rodeheffer, C. A. Thekkath, and D. C. Anderson, “Smartbridge: A scalable Bridge Architecture,” *Sigcomm 2000*, 2000.

[52] E. W. Dijkstra and C. S. Scholten, “Termination Detection for Diffusing Computations,” *Information Processing Letters*, vol. 11(1), pp. 1–4, August 1980.

[53] T. Rodeheffer and M. Schroeder, “Automatic Reconfiguration in Autonet,” tech. rep., SRC Research, Sept. 1991.

[54] K. Lui, W. Lee, and K. Nahrstedt, “STAR: A Transparent Spanning Tree Bridge Protocol with Alternate Routing,” *ACM SIGCOMM 2002*, July 2002.

[55] R. Perlman, “Rbridges: Transparent Routing,” *IEEE Infocom 2004*, 2004.

[56] R. Perlman, S. Gai, and D. G. Dutt, “Rbridges: Base Protocol Specification,” *IETF Draft (work in progress)*, March 2007.

[57] D. C. Plummer, “An Ethernet Address Resolution Protocol or Converting Network Protocol Addresses to 48-bit Ethernet Address for Transmission on Ethernet Hardware,” *IETF RFC 826*, November 1982.

[58] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, “Achieving sub-second IGP convergence in large IP networks,” *ACM SIGCOMM Computer Communication Review*, vol. 35, Issue 3, pp. 35–44, July 2005.

[59] IEEE, “802.1Qay- Provider Backbone Bridge Traffic Engineering,” *IEEE PAR*, March 2007.

[60] T-Pack, “PBT: Carrier Grade Ethernet Transport,” *White Paper*, 2006.

[61] T-Pack, “Transport MPLS: A new Route to Carrier Ethernet,” *T-Pack White Paper*, 2006.

[62] P. Klein and N. Sprecher, “Provider Ethernet VLAN Cross Connect,” vol. 2006, January.

[63] IEEE, “802.1ah -Provider Backbone Bridges (Draft 3.4),” tech. rep., IEEE, March 2007.

[64] IEEE, “802.1ag - Connectivity Fault Management, Draft 8,” March 2007.

[65] IEEE, “802.1qaw - Management of data-driven and data-dependent connectivity faults (draft 0.0),” February 2007.

[66] N. Sprecher, D. Berechya, F. Lingyuan, and J.Liu, “GMPLS Control of Ethernet VLAN Cross Connect Switches,” *IETF Draft*, March 2006.

[67] L. berger (Editor), “Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description,” *IETF RFC 3471*, January 2003.

[68] ITU-T, “G.8110.1/y.1370.1 Architecture of Transport MPLS (T-MPLS) Layer Network,” *available at http://www.itu.int/itudoc/itu-t/aap/sg15aap/recaap/g.8110.1/index.html*.

[69] A. Kirstädter, C. Gruber, J. Riedl, and T. Bauschert, “Carrier-grade Ethernet for packet core networks,” vol. 6354, Oct. 2006.

Rute Sofia graduated (95) in Informatics Engineering from the University of Coimbra; M.Sc. (98) and Ph.D. (2004) in Informatics from the University of Lisbon. During 2000-2003 she was a Visiting Scholar at the Internet Center for Advanced Internet Research (ICAIR, http://www.icair.org), Evanston, USA, and a Visiting Scholar at the University of Pennsylvania, USA. Currently, she is responsible for the coordination of the Internet Architectures and Networking (IAN) of UTM in INESC PORTO. Before joining INESC Porto, she was (2004-2007) a senior research scientist in Siemens AG Corporate Technology/Nokia-Siemens Networks GmbH & Co. KG., focusing on Future Internet topics such as global mobility across multi-access networks (e.g. Mobile IP, WiMAX, 3G) and novel forwarding paradigms (e.g. frame routing, network coding). She was actively involved in the Global Grid Forum and is a contributor to the IETF, as well as member of the IEEE.